MEDNARODNA
PODIPLOMSKA ŠOLA
JOŽEFA STEFANA

INFORMATION AND COMMUNICATION TECHNOLOGIES
PhD study programme

Data Mining and Knowledge Discovery

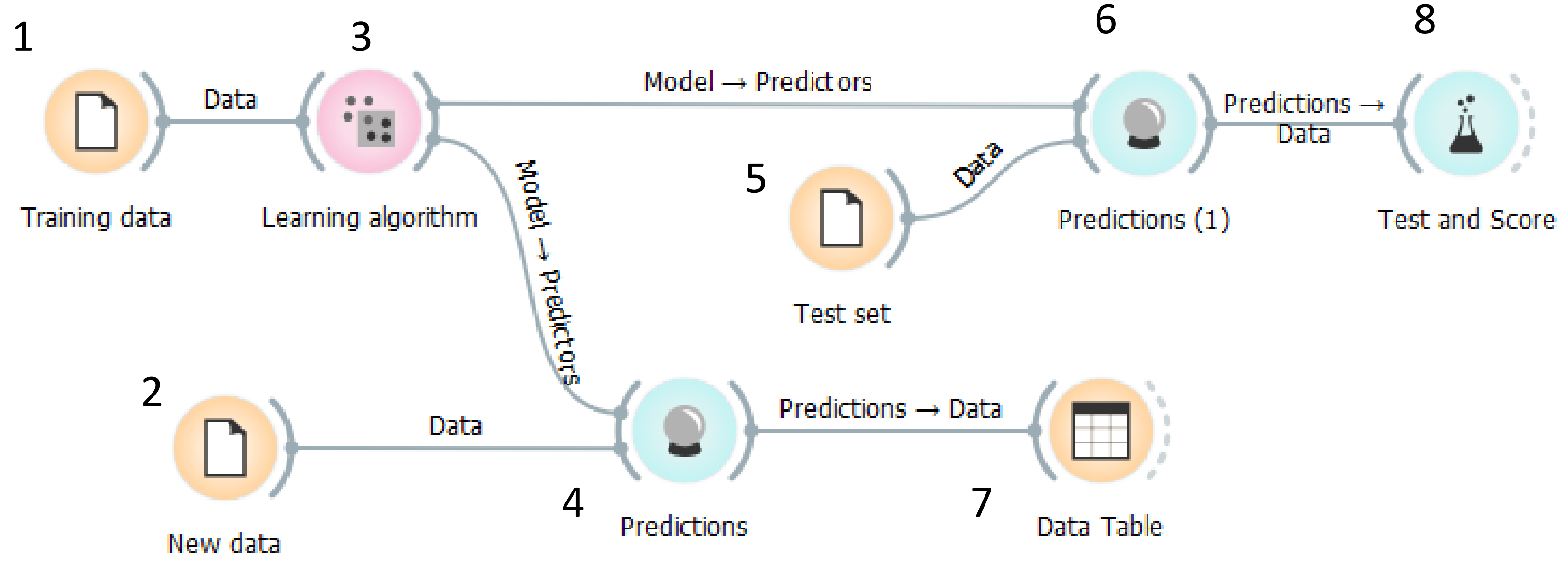
Petra Kralj Novak

December 11, 2019

http://kt.ijs.si/petra_kralj/dmkd3.html

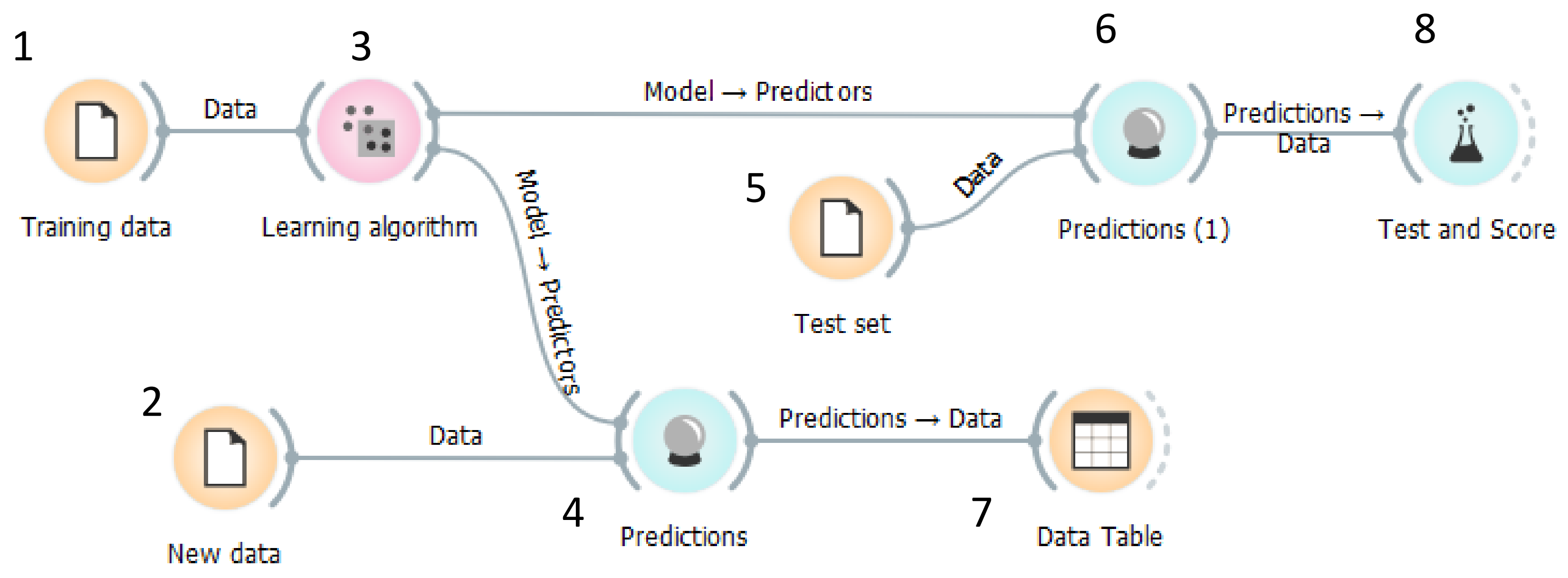
Classification

1. Train the model on train data
2. Test the model on test data
3. Classify new data with the model



Classification

1. Train the model on train data: 1, 3
2. Test the model on test data: 5, 6, 8
3. Classify new data with the model: 2, 4, 7

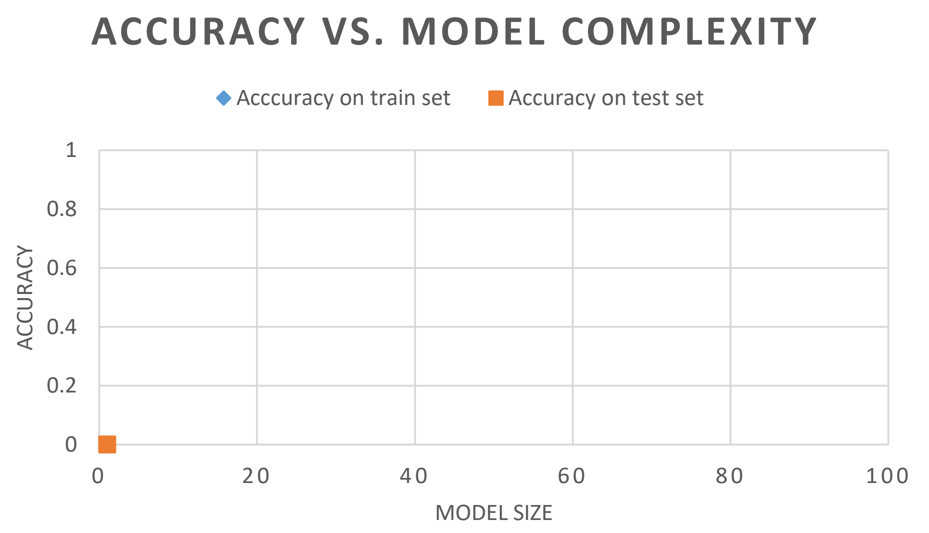


Homework

Model complexity vs. accuracy on train and test set

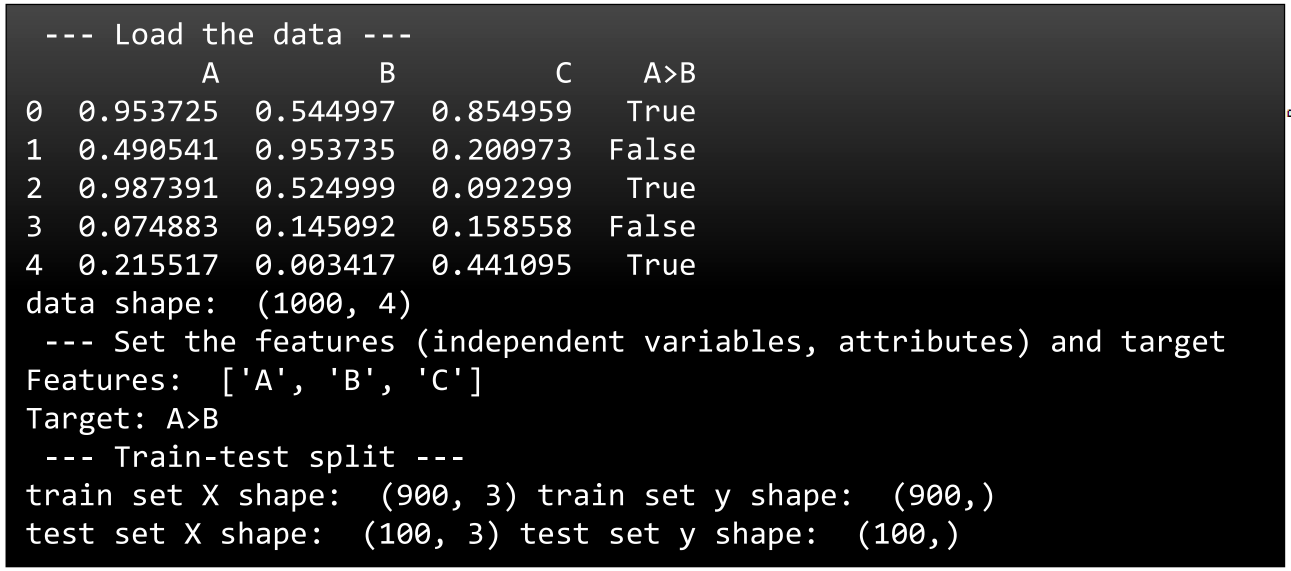
Datasets:

- A-greater-then-B.csv
- Another reasonably sized classification dataset from http://file.biolab.si/datasets/

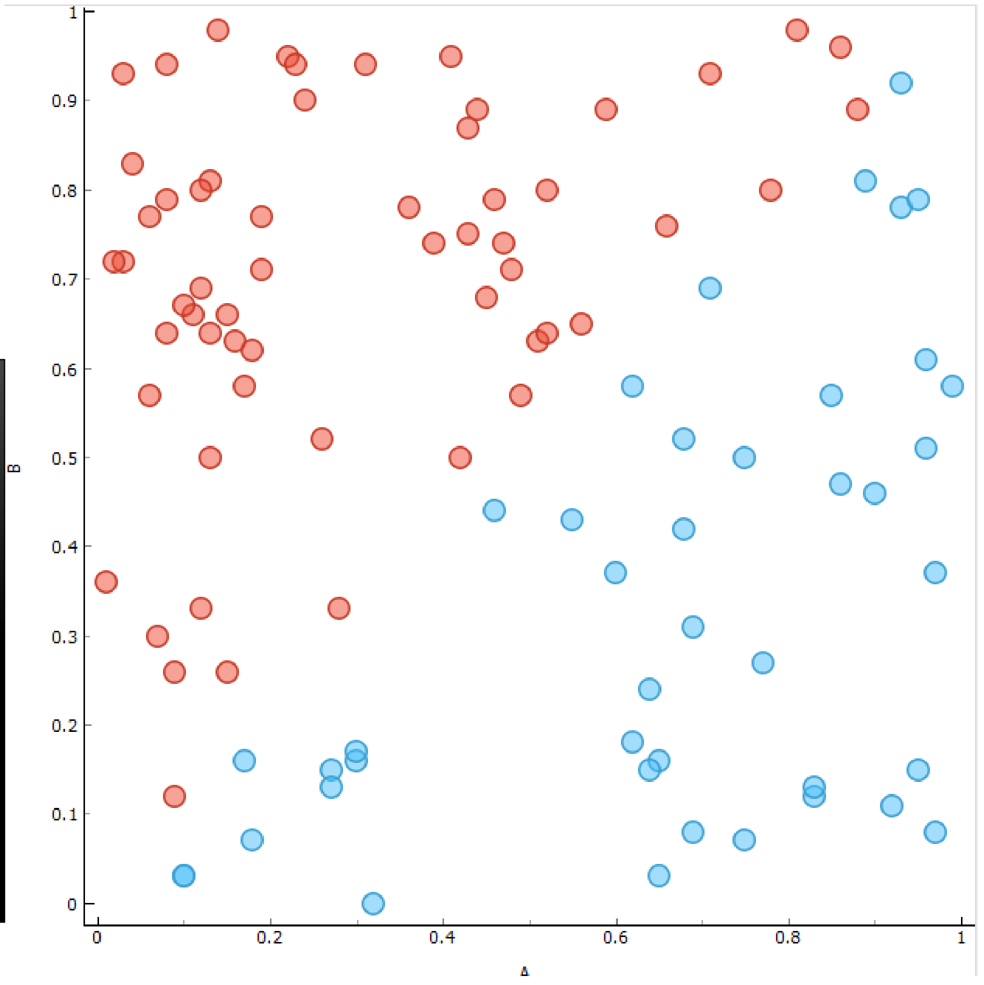


Dataset: A-greater-then-B.csv

- 1000 examples:
    - Attributes A, B and C - random values
    - Target variable „A>B“: „true“ if A>B else “false”

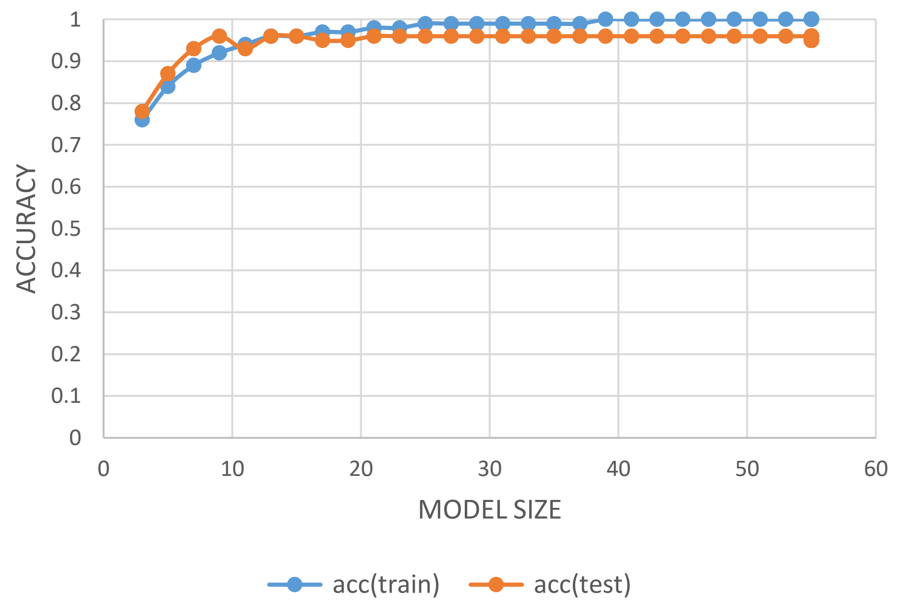


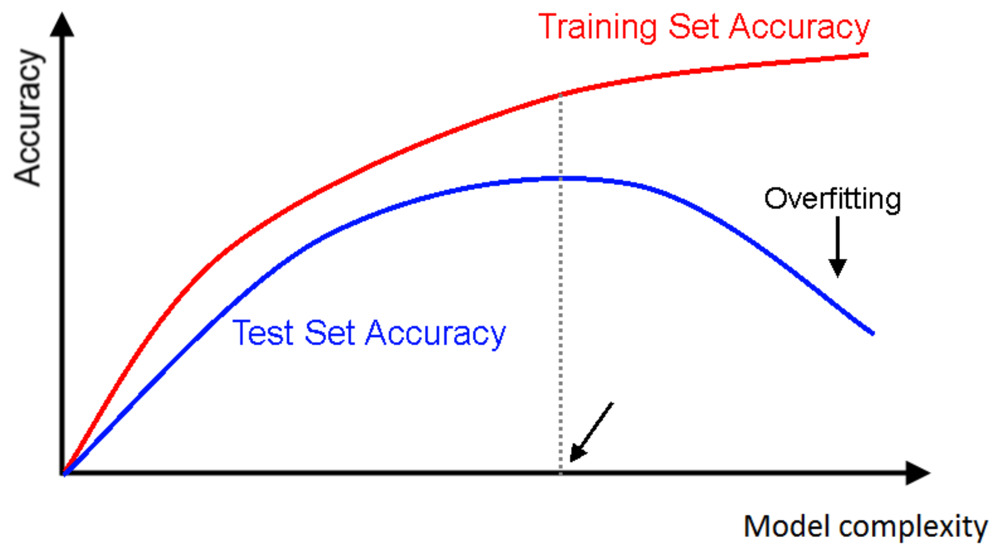
```
 --- Load the data ---
          A         B         C    A>B
0  0.953725  0.544997  0.854959   True
1  0.490541  0.953735  0.200973  False
2  0.987391  0.524999  0.092299   True
3  0.074883  0.145092  0.158558  False
4  0.215517  0.003417  0.441095   True
data shape:  (1000, 4)
 --- Set the features (independent variables, attributes) and target
Features:  ['A', 'B', 'C']
Target: A>B
 --- Train-test split ---
train set X shape:  (900, 3) train set y shape:  (900,)
test set X shape:  (100, 3) test set y shape:  (100,)
```



Accuracy w.r.t. model size

A-greater-then-B.csv





Fitting and overfitting

- Why the accuracy on the test set does not drop?
- How do we know if the model is overfitting?
- Can we use the test set to check for overfitting?

Data Leakage in Machine Learning

- Data leakage refers to a mistake in which information is *accidentally* shared between the test and training data-sets.
- The test set's purpose is to simulate real-world, unseen data.
- Data leakage often results in unrealistically-high levels of performance on the test set, because the model is being ran on data that it had already seen — in some capacity — in the training set.

https://towardsdatascience.com/data-leakage-in-machine-learning-10bdd3eec742

Causes of Data Leakage

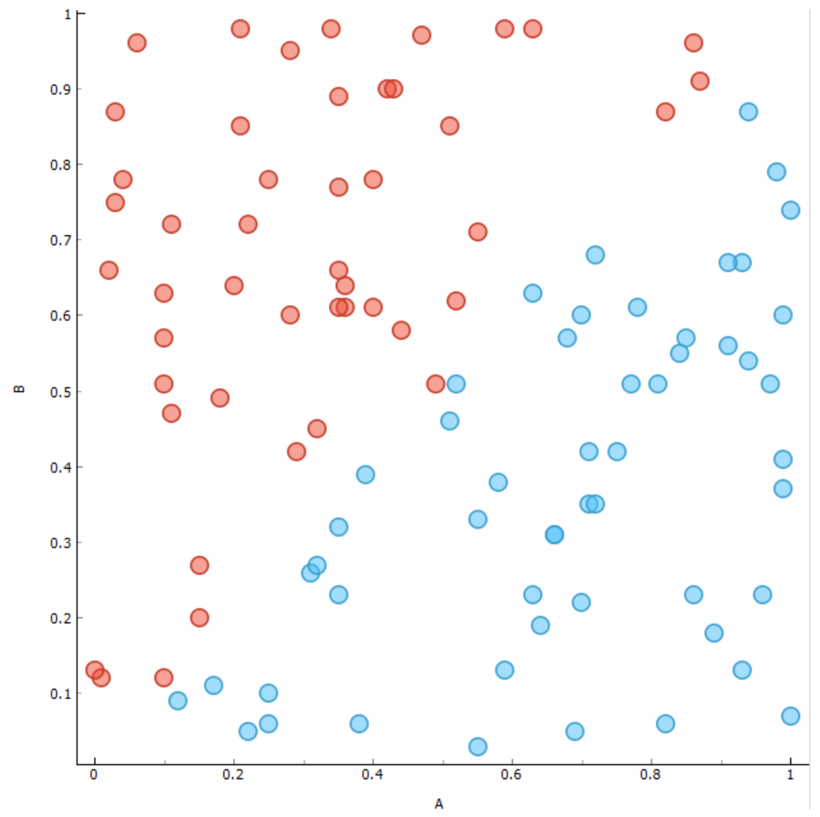
- Pre-processing
  - Feature selection
  - Discretization
  - Missing values imputation
- Duplicates
- Implicit leakage

https://towardsdatascience.com/data-leakage-in-machine-learning-10bdd3eec742

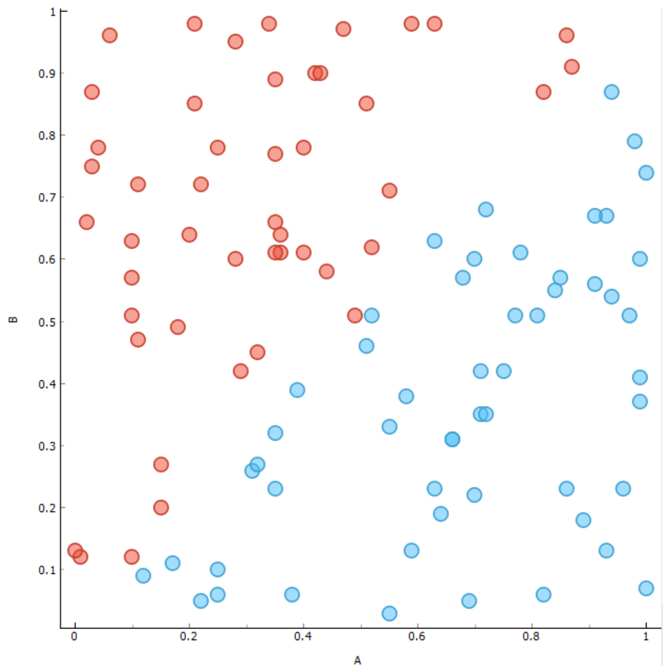
Decision boundary

The boundary between different classes or decision regions is termed as the **decision boundary.**

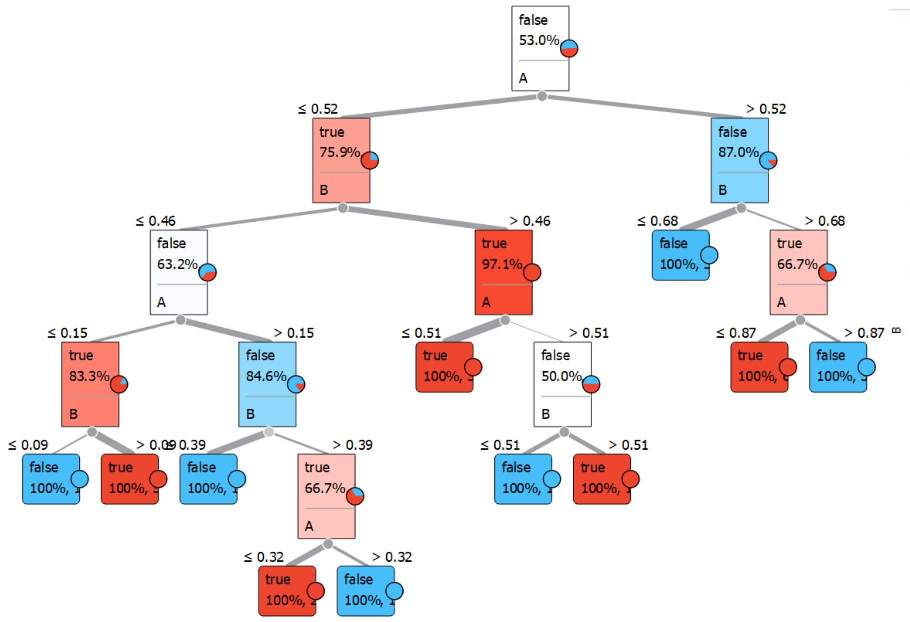
- What is the decision boundary of the model that has generated the data?
- What is a decision boundary like for a decision tree?



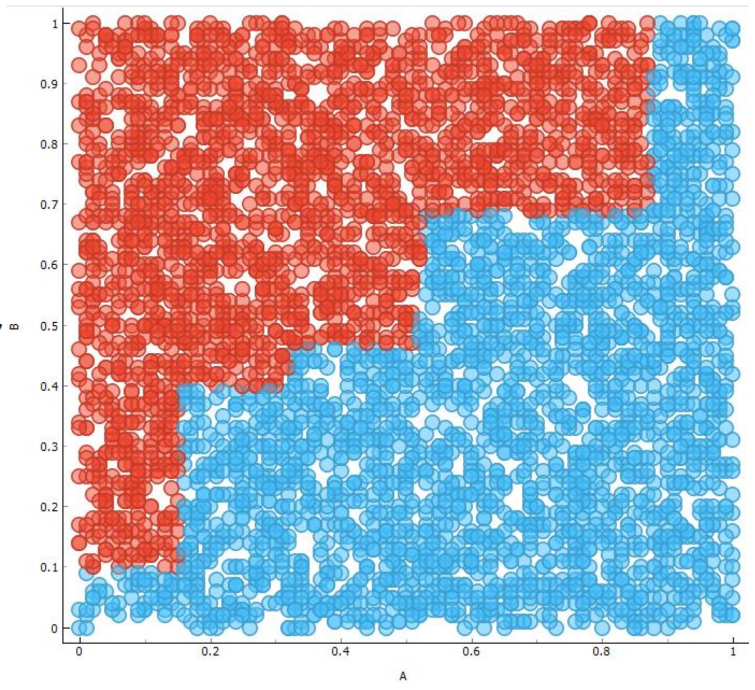
Lab exercise: Decision trees & Language bias



Training set

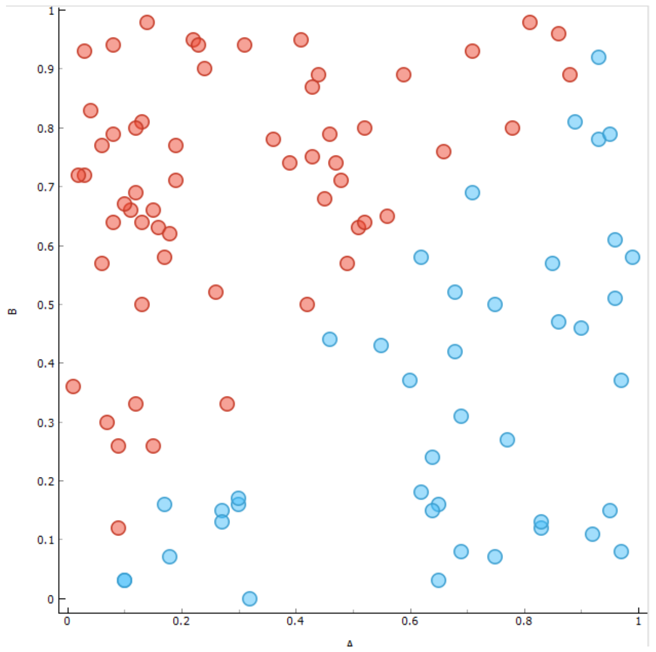


Decision tree

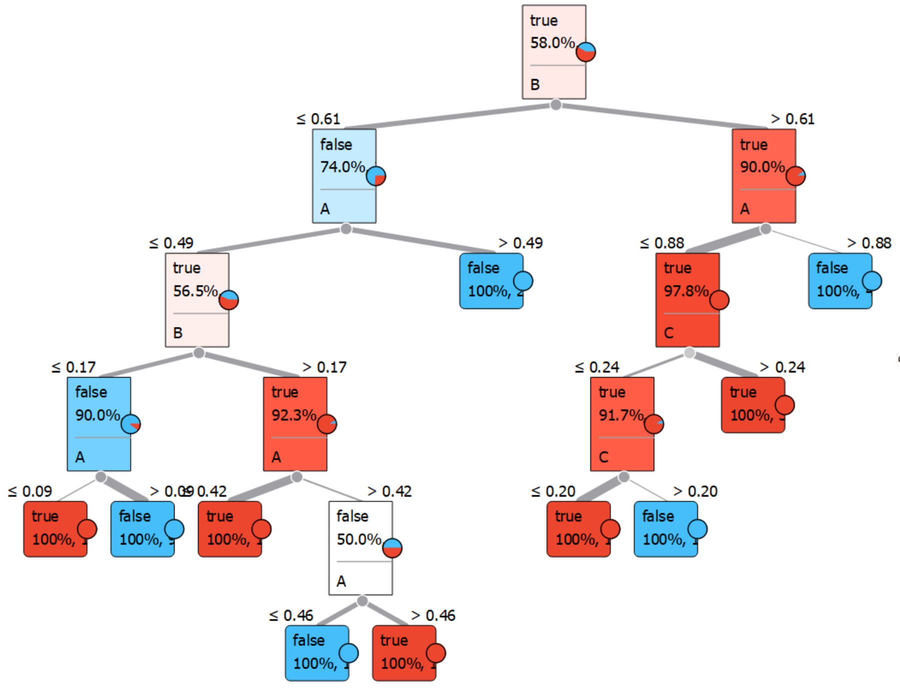


Test set

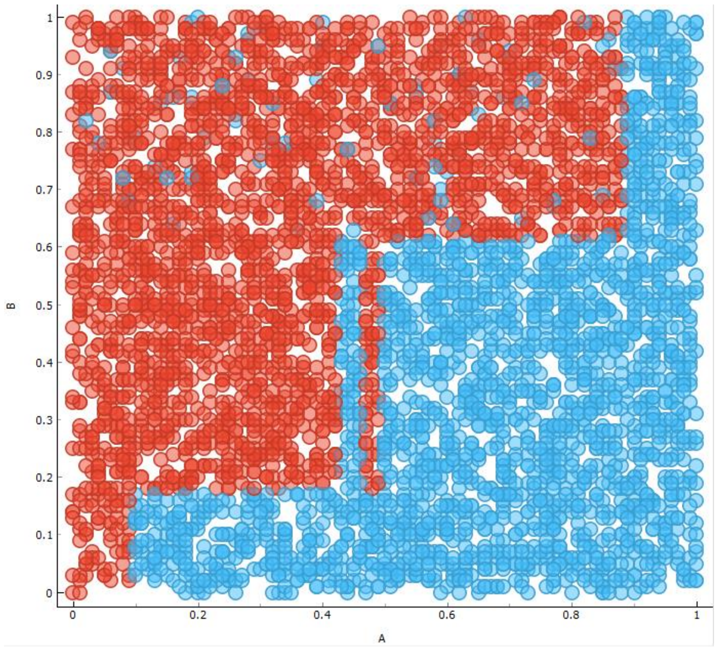
Same program, different random seed



Training set



Decision tree



Test set

How to overcome this

- Feature engineering
  - Create a new feature A>B
  - Examples
    - We have a person’s height and body mass

      → Create a new attribute BMI (bod mass index)

$$BMI = \frac{Weight\ (kg)}{[Height(m)]^2}$$

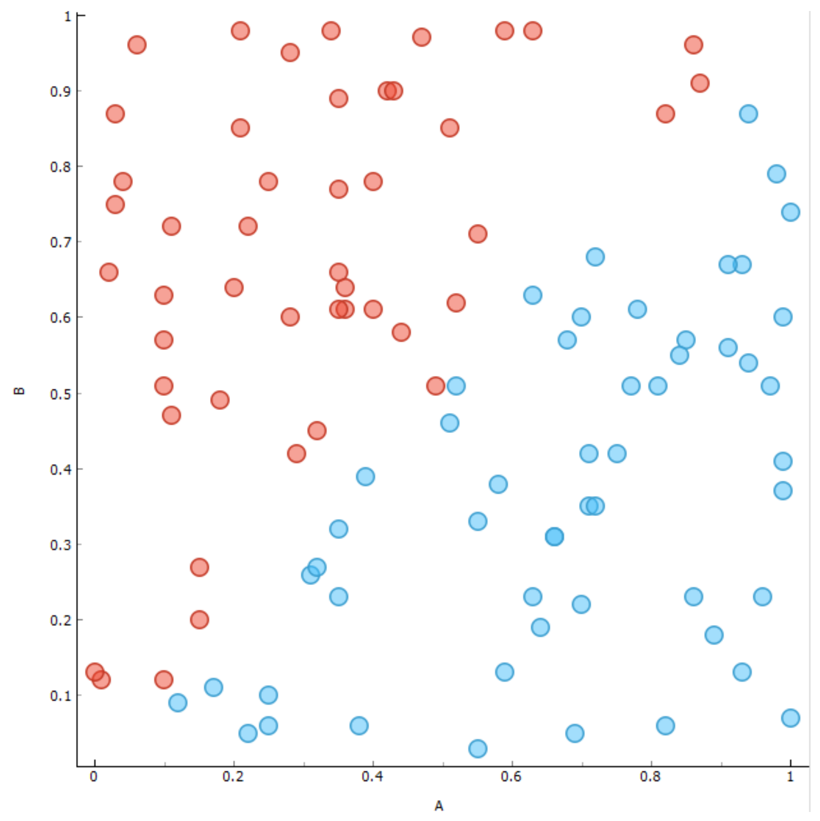
    - We have income and outcome data

      → Create a new attribute “profit”
- Ensemble
  - We build more models that vote for the final classification
  - Random forest: Several trees built on different subsets od the training set
  - On the “A>B” example, decision trees achieve CA 88,2% while random forest 90,8%
  - As a general rule, classifier ensembles always outperform single classifiers
- Use other classifiers
  - Linear classifier, SVM with linear kernel….

Homework

- What is a decision boundary like for KNN?
  - K=1
  - K=3
  - K=10

This can be done by hand, in Orange or in SciKit.



!!!

Evaluation

How good is the model

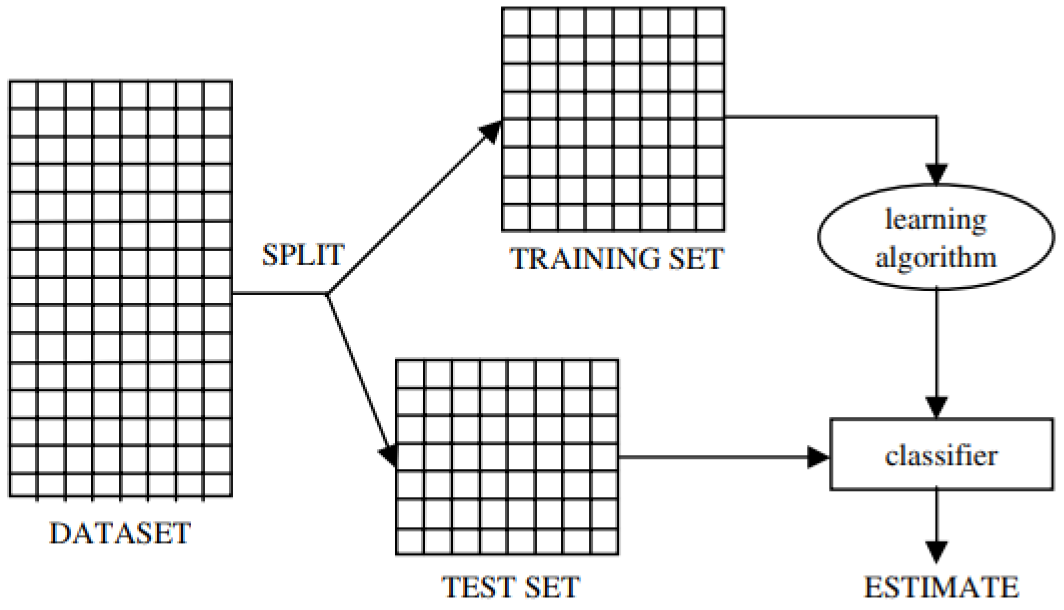
Evaluation goal

- How good is the model

- Method
  - HOW we measure?

- Metric
  - WHAT we measure?

Method: Test on a separate test set



Stratified sampling

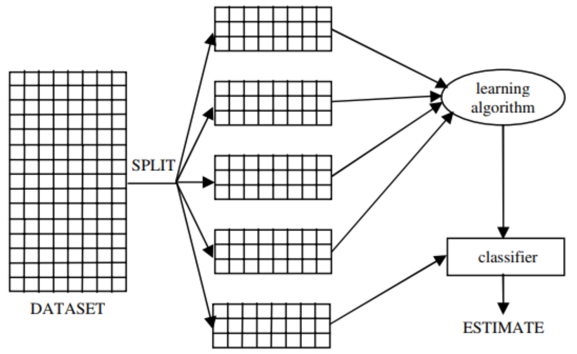
- Stratified sampling aims at splitting one data set so that each split are similar with respect to the target variable distribution.

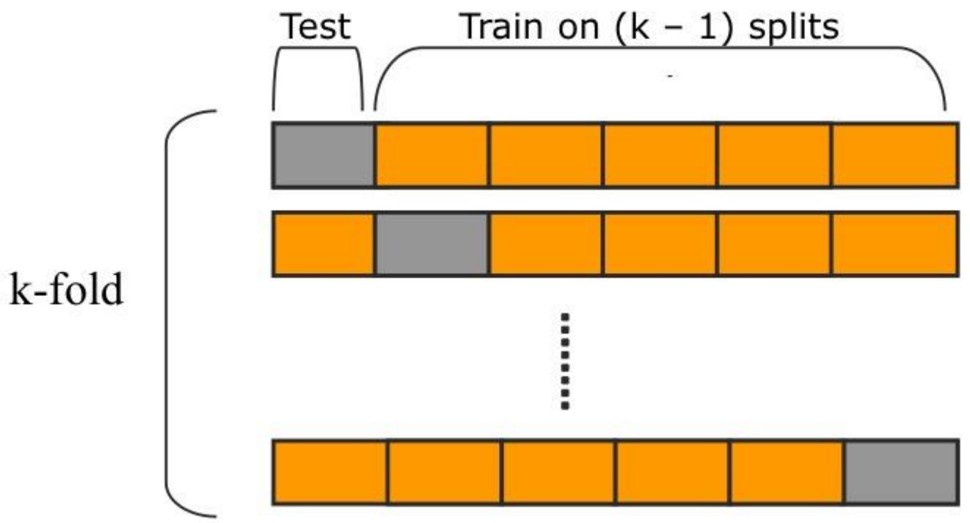
Method: Random sampling

- Repeat several times „Test on a separate test set“ with different test set selections
- Compute the mean, variance on the results …
- The evaluation is more robust as it does not depend so much on a single random split

Method: *K*-fold cross validation

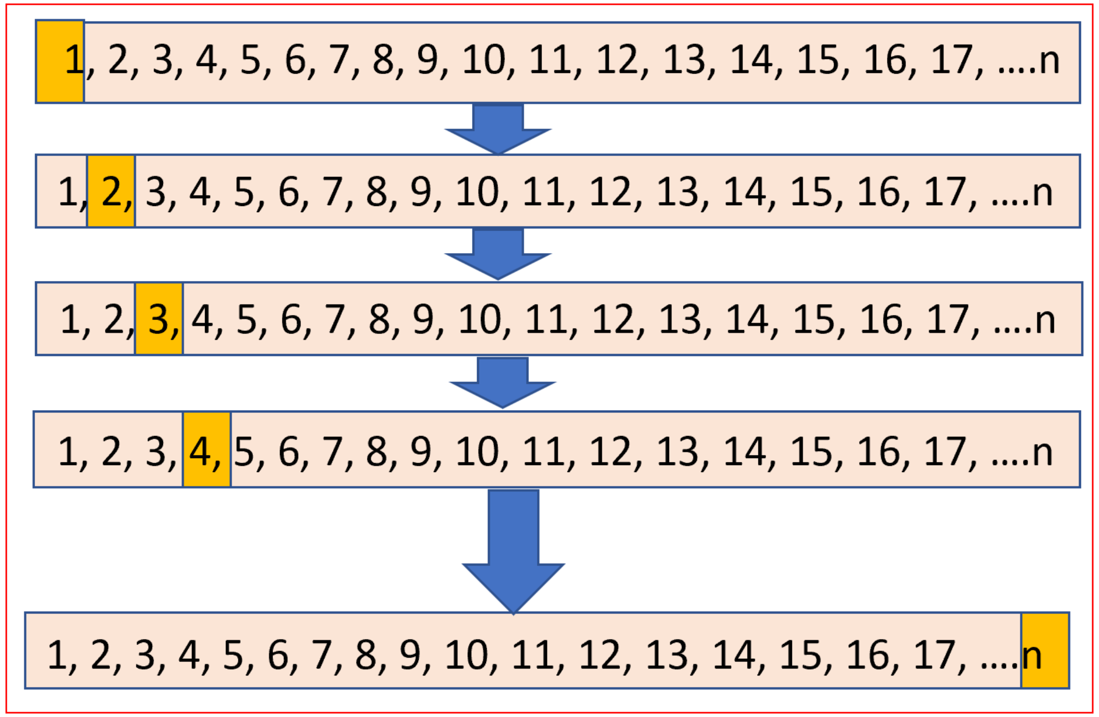
- Most commonly used in machine learning
- Split the dataset into *k* (disjunctive) subsets
- Repeat *k*-times:
  - Use a different subset for testing
  - Use all the other data for training
- Each example is in the test set just once





Method: Leave one out (N-fold cross-validation)

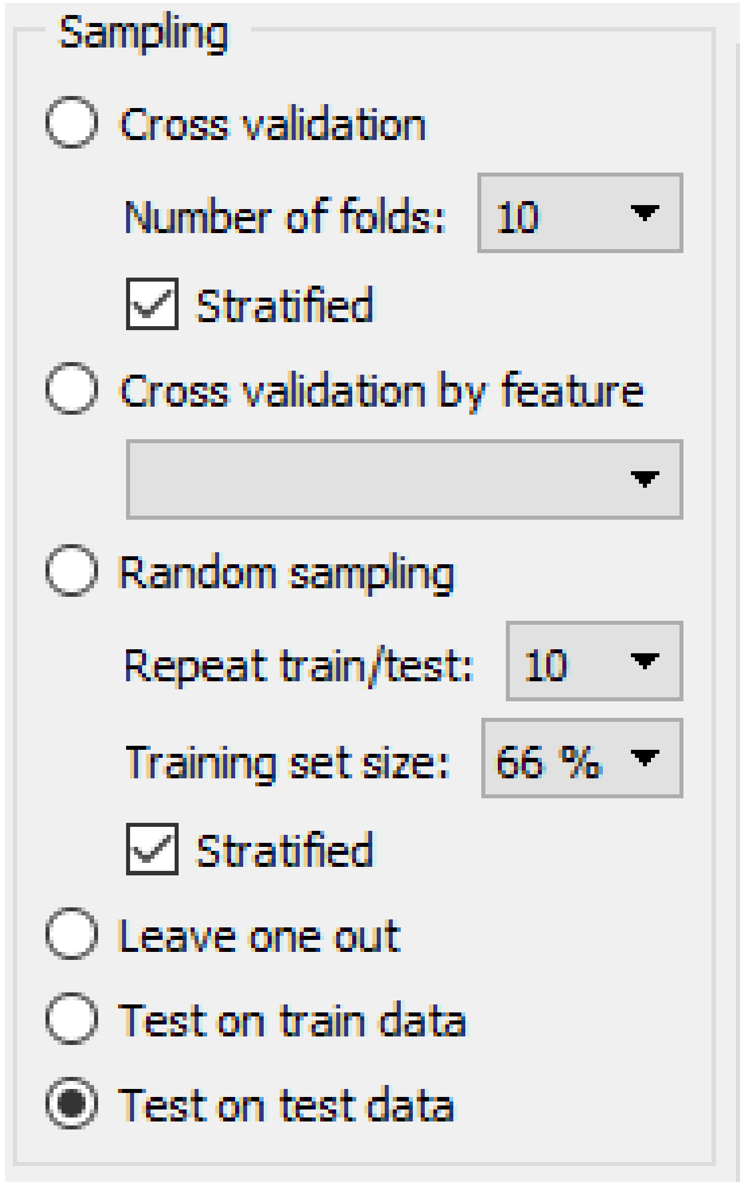
- For small datasets
- Similar to cross validation with test set size =1
- Repeat the training *N*-times if there is *N* examples in the dataset


1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, ….n
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, ….n
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, ….n
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, ….n
1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, ….n

Evaluation methods in Orange

Test & Score

- Cross validation
- Random sampling
- Leave one out
- Test on train data
- Test on test data



Sampling

- ○ Cross validation
  - Number of folds: 10
  - [✓] Stratified
- ○ Cross validation by feature
- ○ Random sampling
  - Repeat train/test: 10
  - Training set size: 66 %
  - [✓] Stratified
- ○ Leave one out
- ○ Test on train data
- ● Test on test data

Questions

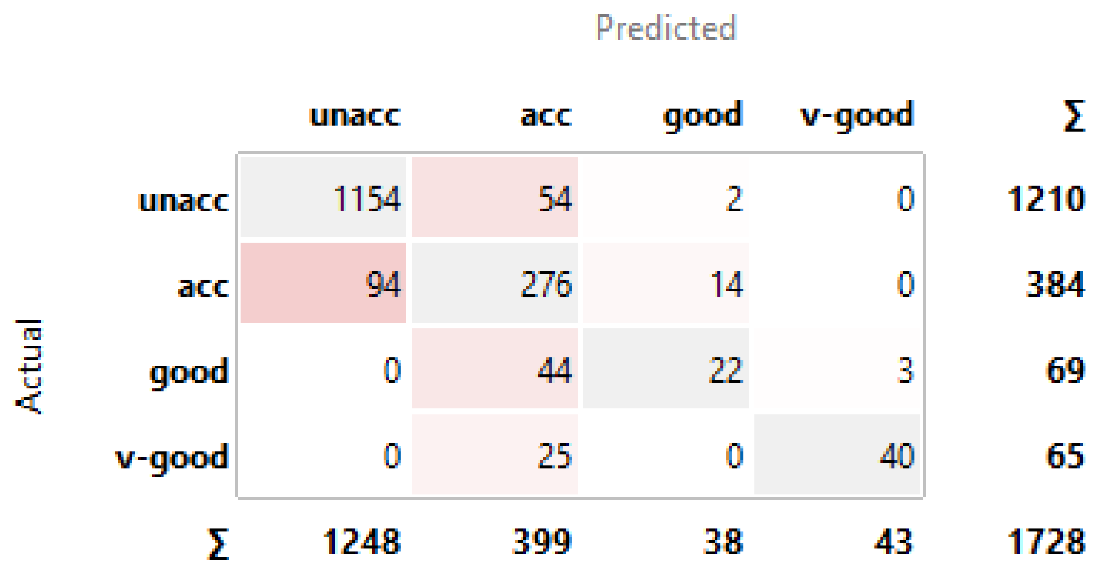
- What are properties of the results of testing on the training set?

Classification quality measures

Confusion matrix (error matrix)

Breakdown of the classifier’s performance, i.e. how frequently instances of class X were correctly classified as class X or misclassified as some other class.

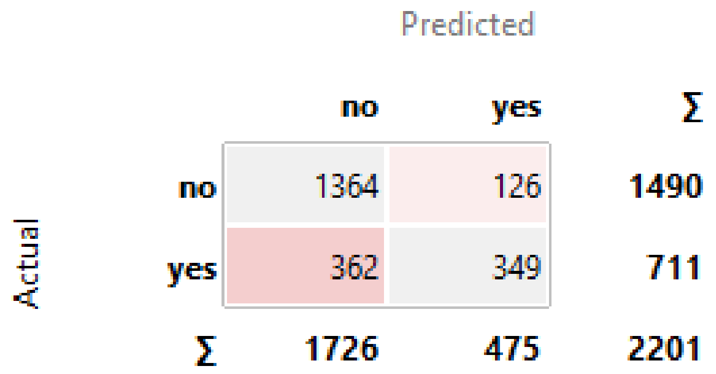
Primer: car



Predicted

Actual	unacc	acc	good	v-good	Σ
unacc	1154	54	2	0	1210
acc	94	276	14	0	384
good	0	44	22	3	69
v-good	0	25	0	40	65
Σ	1248	399	38	43	1728

Primer: titanic

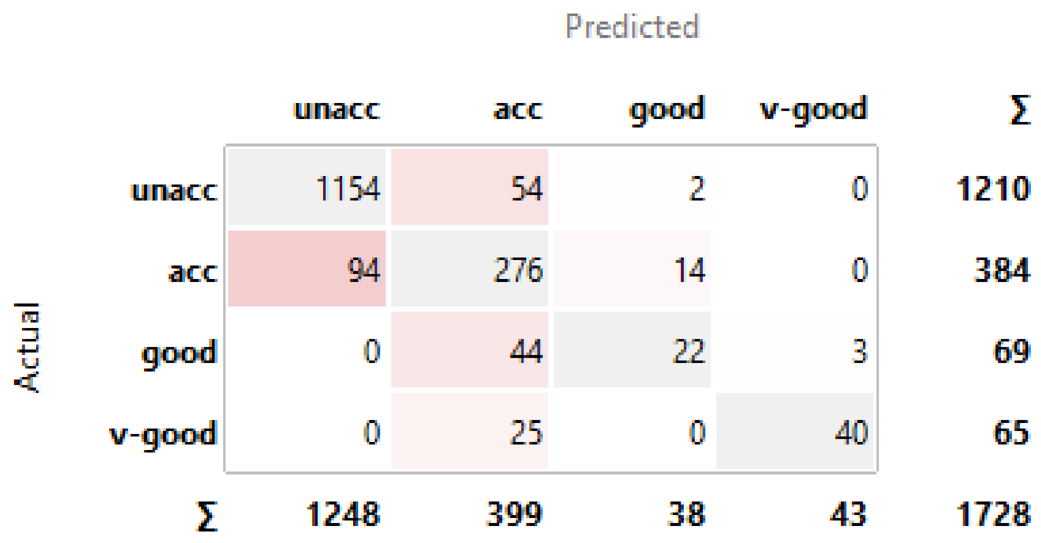


Predicted

Actual	no	yes	Σ
no	1364	126	1490
yes	362	349	711
Σ	1726	475	2201

Confusion matrix

- Matrix of correct and incorrect classifications
  - Rows are actual values
  - Columns are predicted values
  - Correct classifications are on the diagonal



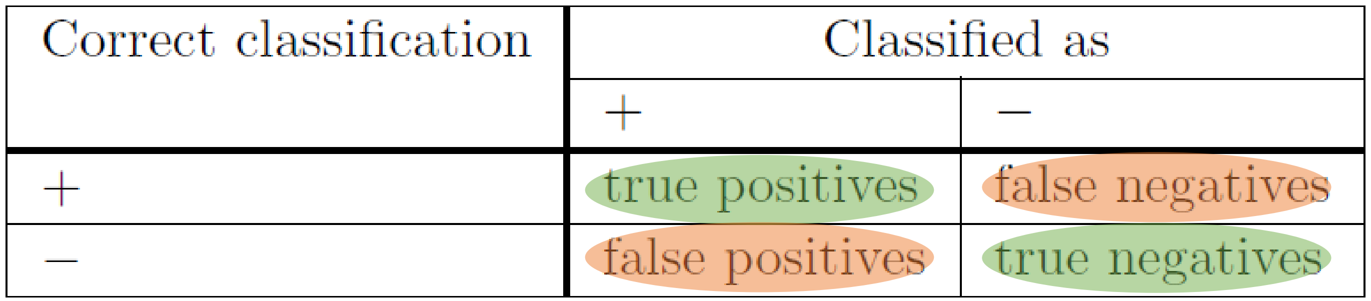
Predicted

Actual	unacc	acc	good	v-good	Σ
unacc	1154	54	2	0	1210
acc	94	276	14	0	384
good	0	44	22	3	69
v-good	0	25	0	40	65
Σ	1248	399	38	43	1728

Confusion matrix for two classes

Predicted

Actual



Correct classification	Classified as	
	+	−
+	true positives	false negatives
−	false positives	true negatives

TP: true positives
The number of positive instances that are classified as positive

FP: false positives
The number of negative instances that are classified as positive

FN: false negatives
The number of positive instances that are classified as negative

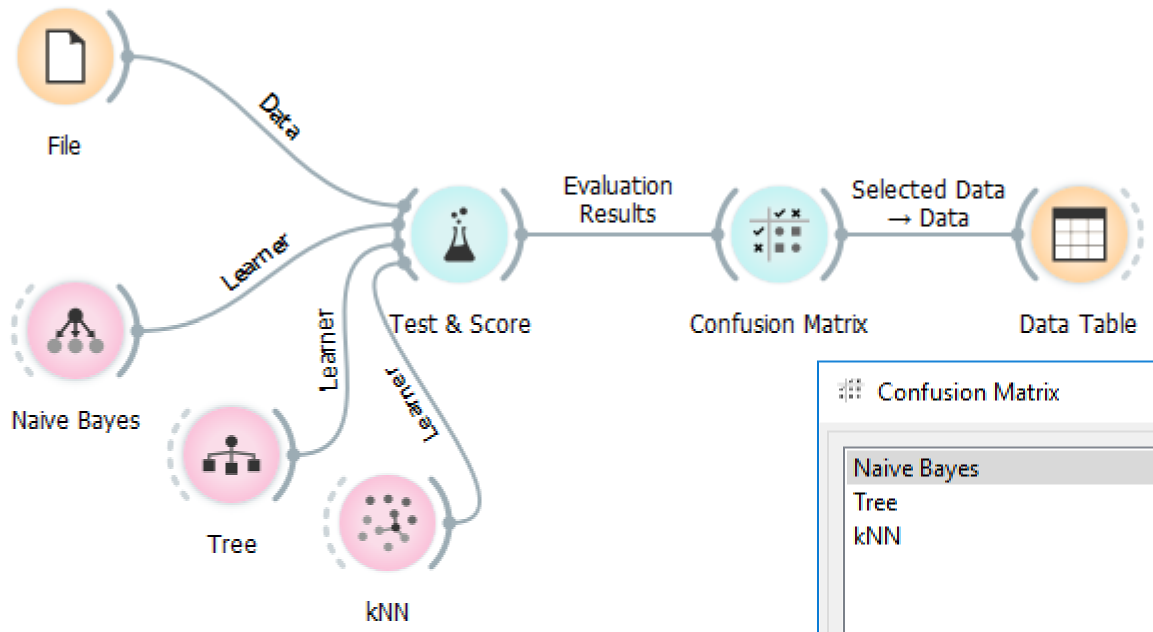
TN: true negatives
The number of negative instances that are classified as negative

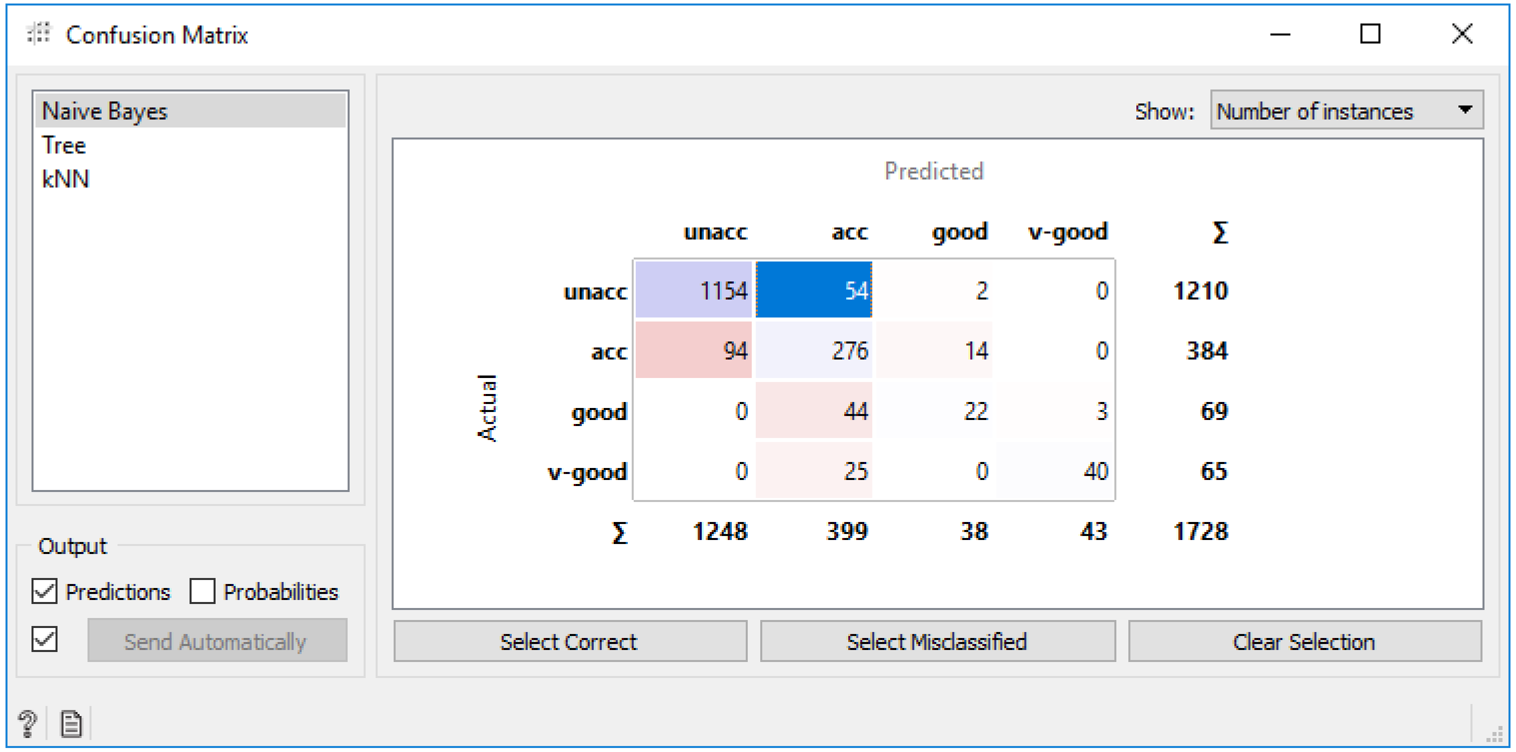
- Diagonal: correct classifications
- Outside: misclassifications
- Classification accuracy =

= |correct classifications| / |all examples|

= |correct classifications| / (|correct classifications| + |misclassifications|)

In Orange, the confusion matrix is interactive





Show: Number of instances

Output: [✓] Predictions [ ] Probabilities

Predicted

Actual	unacc	acc	good	v-good	Σ
unacc	1154	54	2	0	1210
acc	94	276	14	0	384
good	0	44	22	3	69
v-good	0	25	0	40	65
Σ	1248	399	38	43	1728

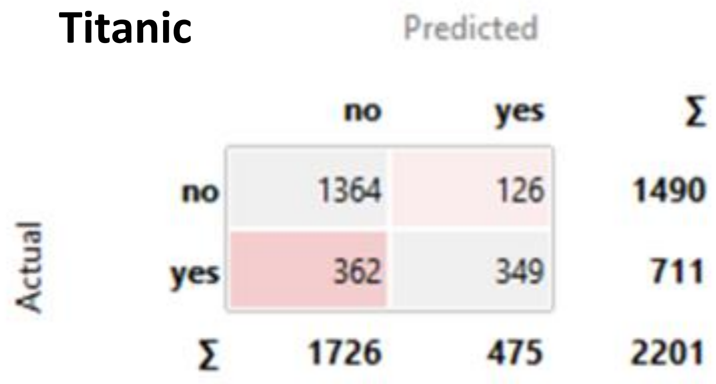
Classification accuracy

- Percentage of correctly classified examples

Classification accuracy =
= |correct classifications| / |all examples|
= |correct classifications| / (|correct classifications| + |misclassifications|)

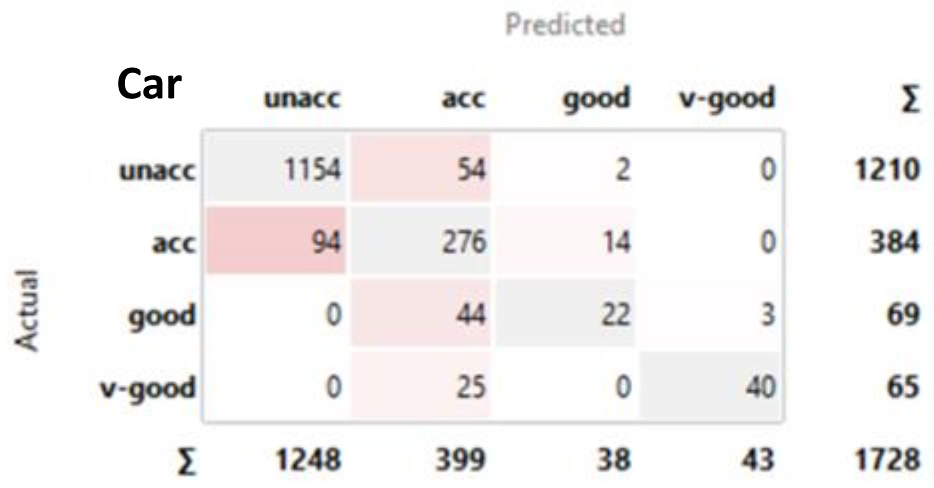
Homework: Confusion matrix

Titanic



Actual \ Predicted	no	yes	Σ
no	1364	126	1490
yes	362	349	711
Σ	1726	475	2201

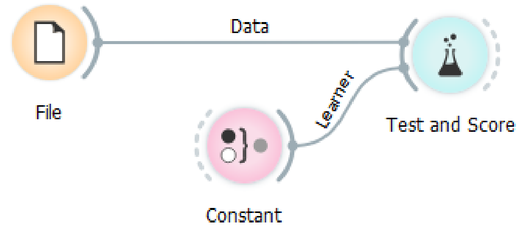
Car

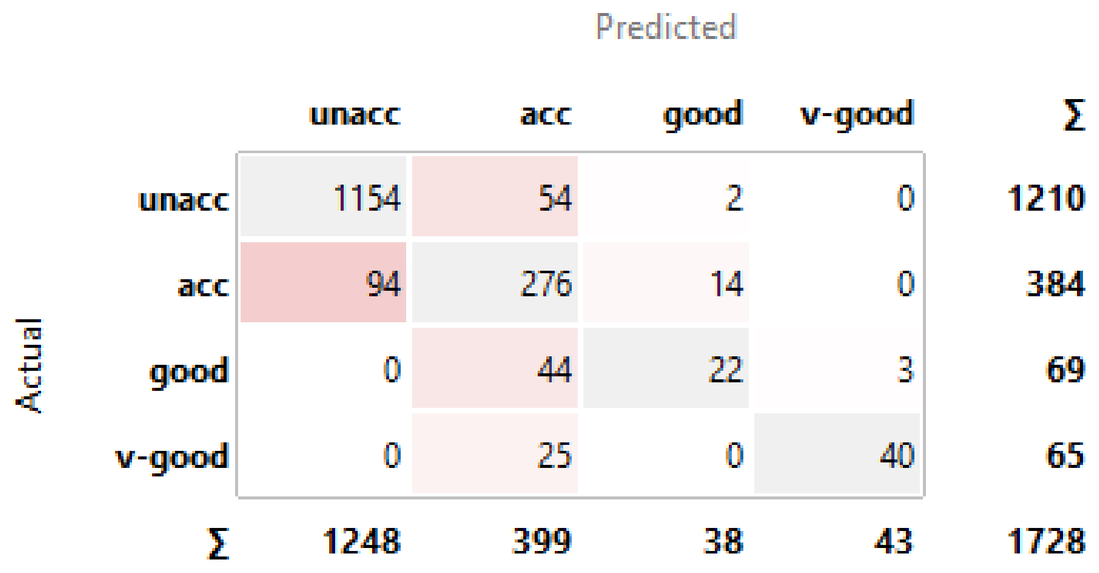


Actual \ Predicted	unacc	acc	good	v-good	Σ
unacc	1154	54	2	0	1210
acc	94	276	14	0	384
good	0	44	22	3	69
v-good	0	25	0	40	65
Σ	1248	399	38	43	1728

	Titanic	Car
Number of examples		
Number of classes		
Number of examples in each class		
Number of examples classified in individual classes		
Number of misclassified examples		
Classification accuracy		

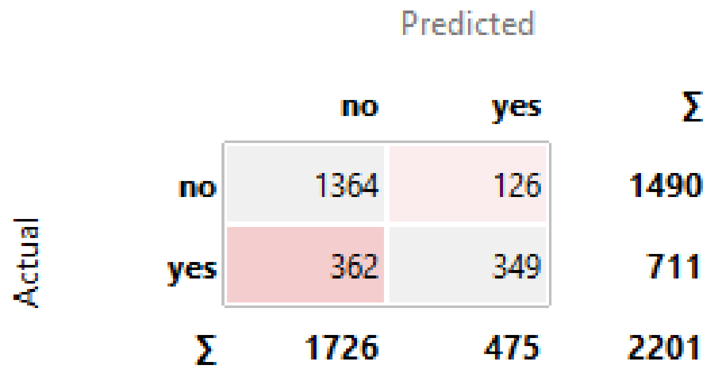
Majority class classifier (Constant)





Predicted

Actual	unacc	acc	good	v-good	Σ
unacc	1154	54	2	0	1210
acc	94	276	14	0	384
good	0	44	22	3	69
v-good	0	25	0	40	65
Σ	1248	399	38	43	1728



Predicted

Actual	no	yes	Σ
no	1364	126	1490
yes	362	349	711
Σ	1726	475	2201

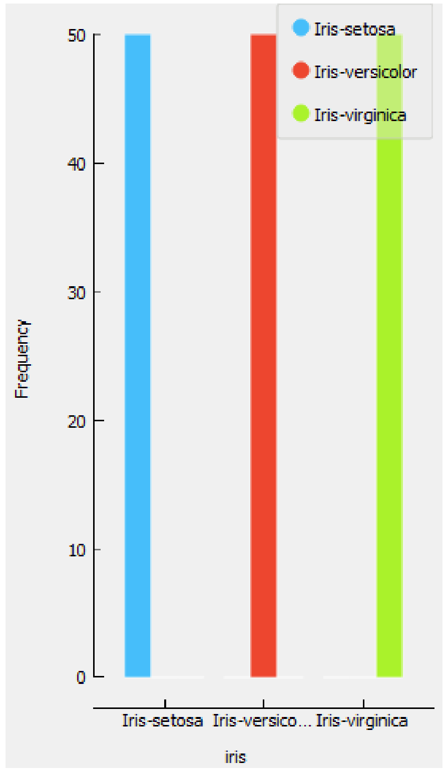
- What is the classification accuracy of a classifier that classifies all the examples in the majority class?
- Car: 70% Titanic: 68%

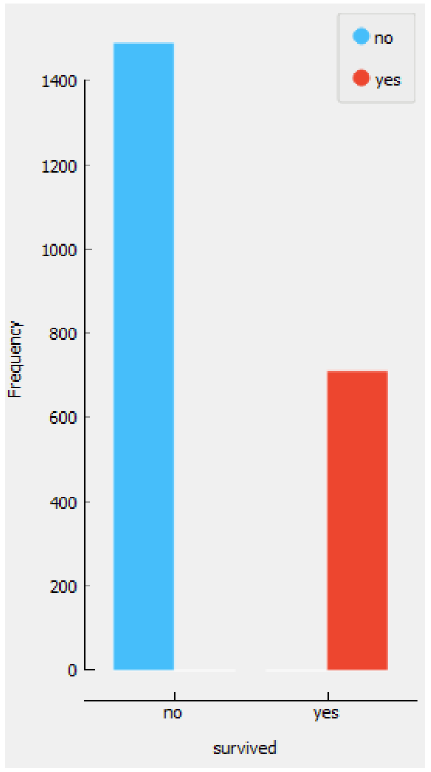
Question

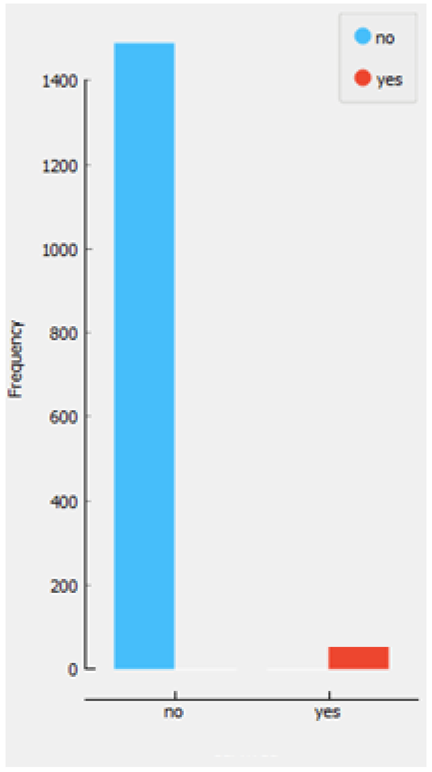
- When is classification accuracy “good”?

Imbalanced Data and Unequal Misclassification Costs

- Imbalanced dataset: One class is minority compared to the other(s)
  - The minority class is usually the one of interest







Imbalanced Data and Unequal Misclassification Costs

- Imbalanced dataset: One class is minority compared to the other(s)
  - The minority class is usually the one of interest
- Unequal misclassification costs:
  - Some errors are more costly (have more severe consequences)
- Examples:
  - Screening tests (nuchal scan, Zora, Dora, Svit, …)
  - Intrusion detection
  - Credit card fraud

Zora
Državni program zgodnjega odkrivanja predrakavih sprememb materničnega vratu

Dora
DRŽAVNI PRESEJALNI PROGRAM ZA RAKA DOJK

Svit
DRŽAVNI PROGRAM PRESEJANJA IN ZGODNJEGA ODKRIVANJA PREDRAKAVIH SPREMEMB IN RAKA NA DEBELEM ČREVESU IN DANKI

Exercise: Credit card fraud

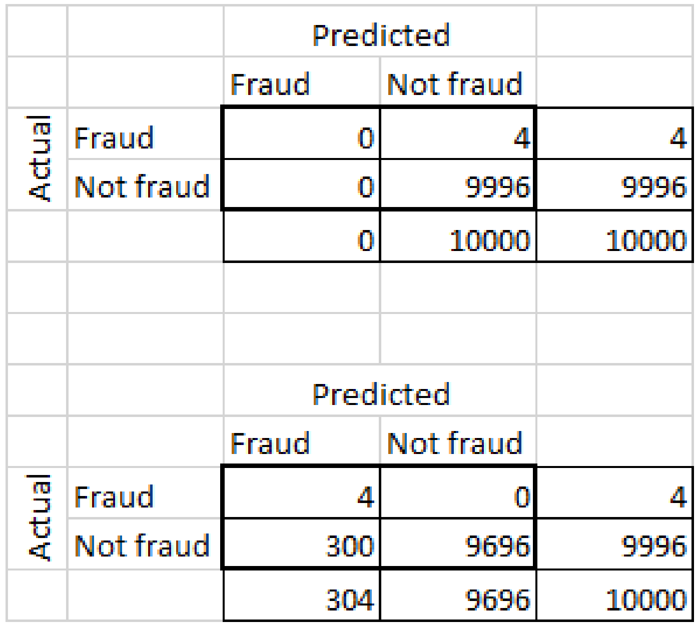
„FED report notes the fraud rate for debit and prepaid signature transactions in 2012 was approximately 4.04 basis points (bps), or about ***four per every 10,000 transactions****.“*

- What is the classification accuracy of a classifier that classifies all the examples a „not fraudulent“?
  - Answer: 99.96%
- Can a classifier with a 97% accuracy “better” then the one with classification accuracy 99.96%?

https://www.pymnts.com/in-depth/2014/a-tale-of-two-fraud-stats/

Exercise: Credit card fraud

Two confusion matrices for two classifiers



		Predicted		
		Fraud	Not fraud	
Actual	Fraud	0	4	4
	Not fraud	0	9996	9996
		0	10000	10000

		Predicted		
		Fraud	Not fraud	
Actual	Fraud	4	0	4
	Not fraud	300	9696	9996
		304	9696	10000

Classification accuracy

- CA = (0 + 9996)/10000 = 99.96%

- CA = (4 + 9696)/10000 = 97.00%

A model with a worse classification accuracy compared to the majority class is better.

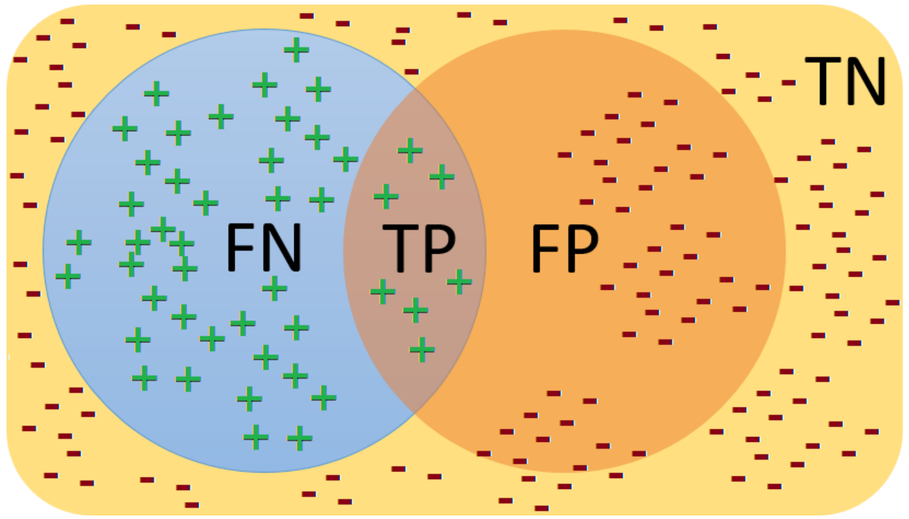
Precision and Recall

PRECISION

- Out of all the examples the classifier labeled as positive, what fraction were correct?

RECALL

- Out of all the positive examples there were, what fraction did the classifier pick up?



Precision, Recall & F1

- Class-specific metrics
  - Precision (Positive Predictive Value)
    - Proportion of instances classified as positive that are really positive
  - Recall (True Positive Rate, TP Rate, Hit Rate, Sensitivity)
    - The proportion of positive instances that are correctly classified as positive
  - F1
    - Harmonic mean of precision and recall

$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

- We can average the metrics over the classes (macro average) or weigh them by the number of examples (micro average)

Precision, recall, F1

		Predicted class		Total instances
		+	−	
Actual class	+	TP	FN	P
	−	FP	TN	N

True Positive Rate or Hit Rate or Recall or Sensitivity or TP Rate	TP/P	The proportion of positive instances that are correctly classified as positive
Precision or Positive Predictive Value	TP/(TP+FP)	Proportion of instances classified as positive that are really positive
F1 Score	(2 × Precision × Recall) /(Precision + Recall)	A measure that combines Precision and Recall
Accuracy or Predictive Accuracy	(TP + TN)/(P + N)	The proportion of instances that are correctly classified

- Priklic
- Natančnost
- Mera F1
- Klasifikacijska točnost

Homework: F1

- Express F1 in terms of TP, FP, TN, FN

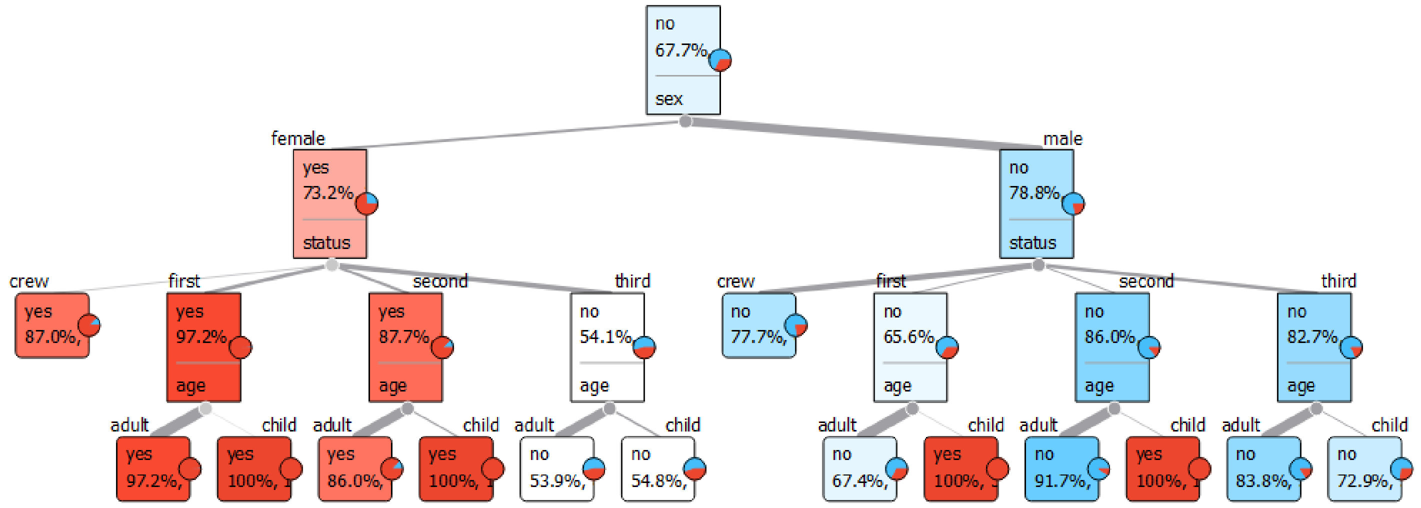
$$F_1 = 2 * \frac{precision * recall}{precision + recall}$$

		Predicted class		Total instances
		+	−	
Actual class	+	TP	FN	P
	−	FP	TN	N

ROC

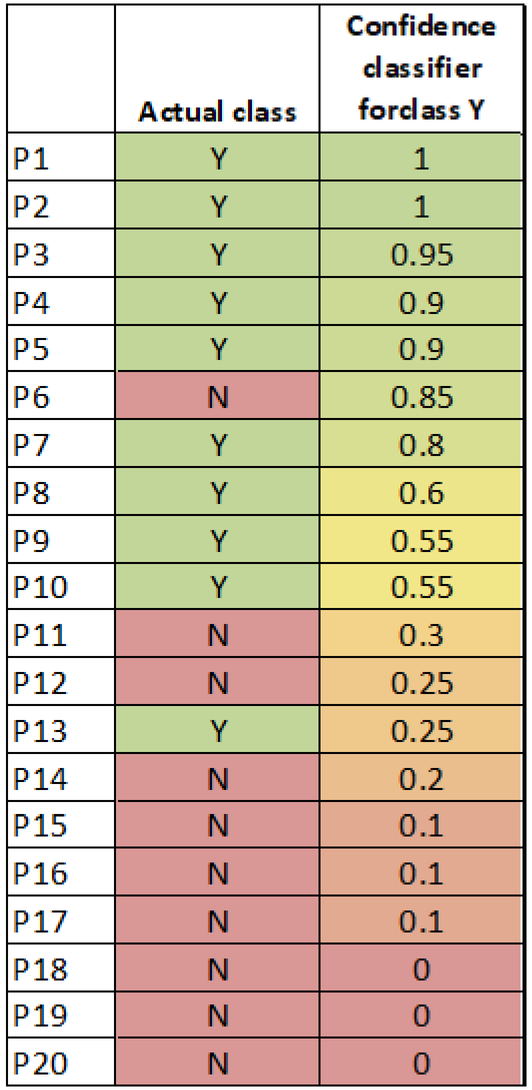
High precision and/or high recall?

- Can we make a model more precise (increase precision)?
- How sure is the model about a certain prediction?
- We can set different thresholds and get different binary classifiers.
- Find a trade-off between precision and recall appropriate for a problem at hand.



Probabilistic classification

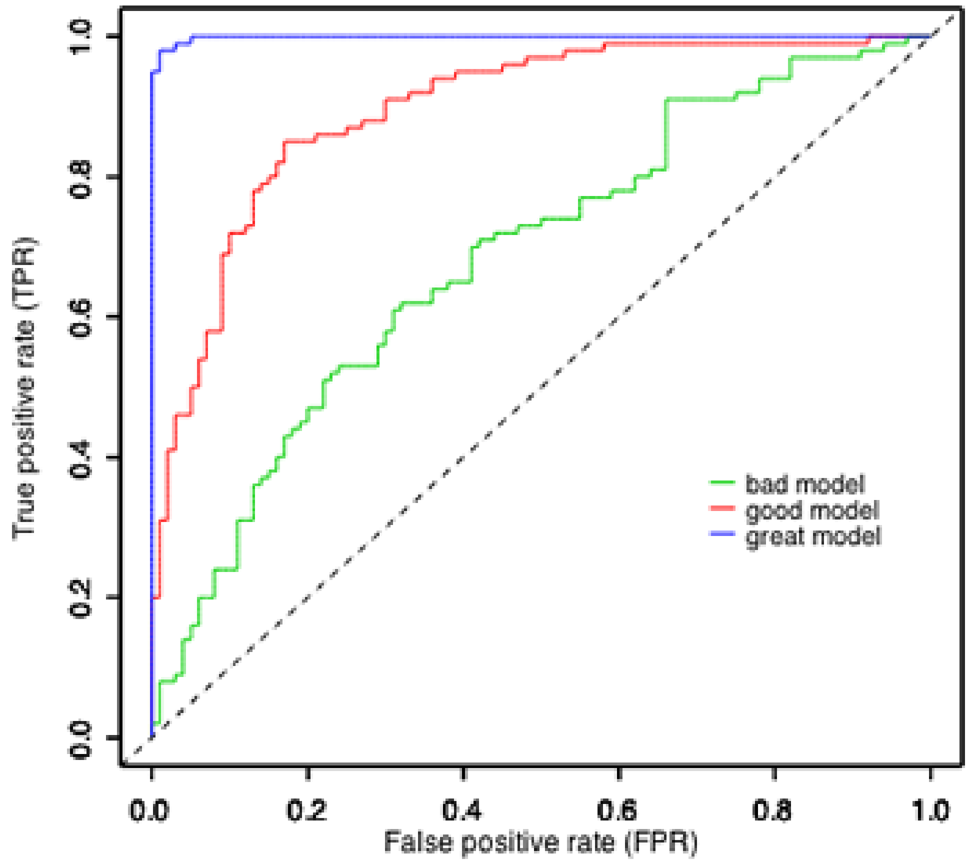
- A **probabilistic** classifier is a classifier that is able to predict, given an observation of an input, a **probability** distribution over a set of classes, rather than only outputting the most likely class that the observation should belong to.
- Ranking
- Tresholds/cutpoints



	Actual class	Confidence classifier for class Y
P1	Y	1
P2	Y	1
P3	Y	0.95
P4	Y	0.9
P5	Y	0.9
P6	N	0.85
P7	Y	0.8
P8	Y	0.6
P9	Y	0.55
P10	Y	0.55
P11	N	0.3
P12	N	0.25
P13	Y	0.25
P14	N	0.2
P15	N	0.1
P16	N	0.1
P17	N	0.1
P18	N	0
P19	N	0
P20	N	0

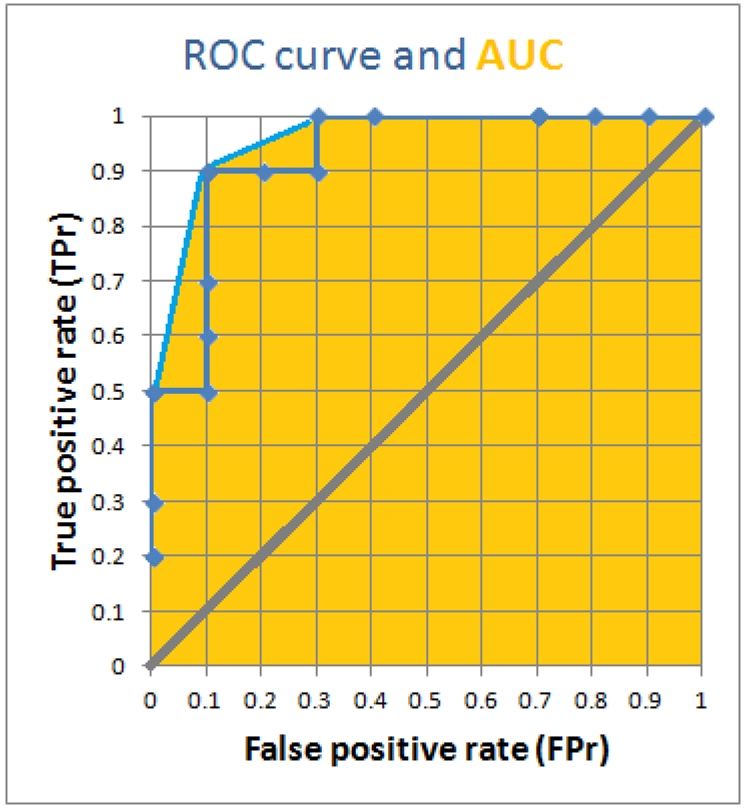
ROC curve and AUC

- **Receiver Operating Characteristic curve** (or ROC curve) is a plot of the true positive rate (TPr=Sensitivity=Recall) against the false positive rate (FPr) for different possible cutpoints.
- It shows the tradeoff between sensitivity and specificity (any increase in sensitivity will be accompanied by a decrease in specificity).
- The closer the curve to the top left corner, the "better" the classifier.
- The diagonal represents the random classifiers (predicting the positive class with some probability regardless the data).



AUC - Area Under (ROC) Curve

- Performance is measured by the area under the ROC curve (AUC). An area of 1 represents a perfect classifier; an area of 0.5 represents a worthless classifier.
- The area under the curve (AUC) is equal to the probability that a classifier will rank a randomly chosen positive example higher than a randomly chosen negative example.

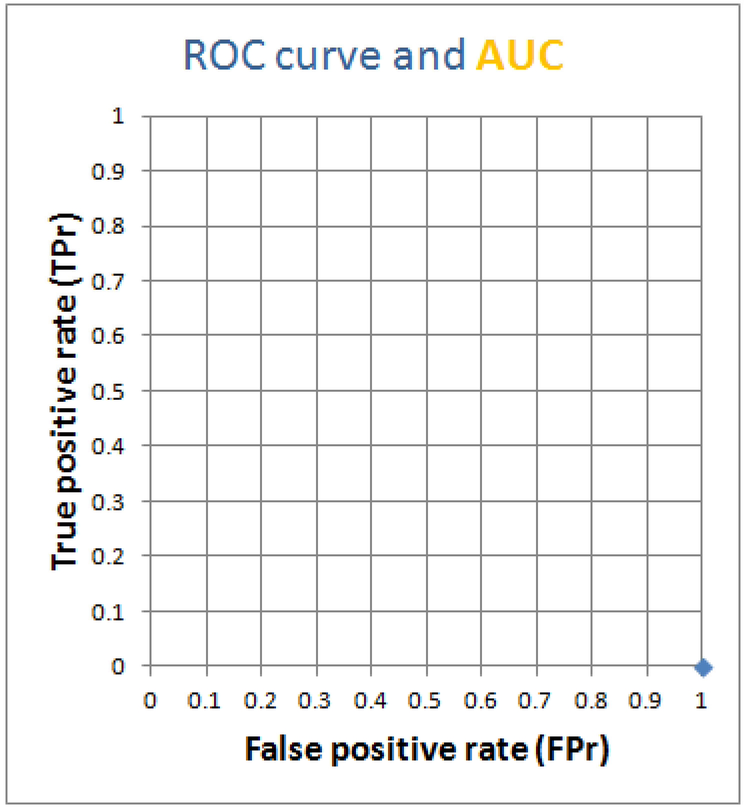


Exercise: ROC curve and AUC

	Actual class	Confidence classifier forclass Y	FP	TP	FPr	TPr
P1	Y	1				
P2	Y	1				
P3	Y	0.95				
P4	Y	0.9				
P5	Y	0.9				
P6	N	0.85				
P7	Y	0.8				
P8	Y	0.6				
P9	Y	0.55				
P10	Y	0.55				
P11	N	0.3				
P12	N	0.25				
P13	Y	0.25				
P14	N	0.2				
P15	N	0.1				
P16	N	0.1				
P17	N	0.1				
P18	N	0				
P19	N	0				
P20	N	0				

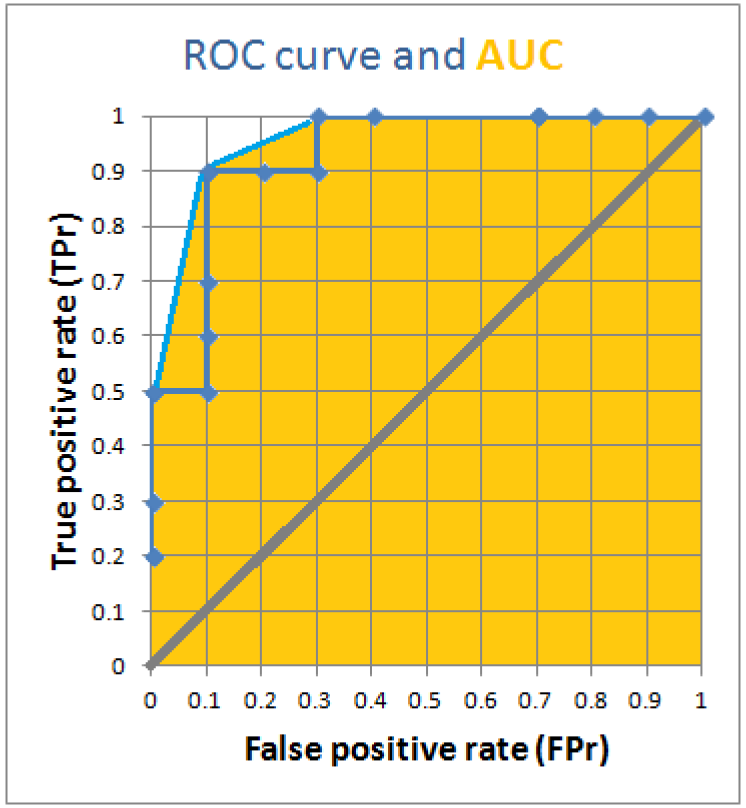
ROC curve and AUC

	Actual class	Classifier confidence forclass Y	FP	TP	FPr	TPr
P1	Y	1	0	2	0	0.2
P2	Y	1	0	2	0	0.2
P3	Y	0.95	0	3	0	0.3
P4	Y	0.9	0	5	0	0.5
P5	Y	0.9	0	5	0	0.5
P6	N	0.85	1	5	0.1	0.5
P7	Y	0.8	1	6	0.1	0.6
P8	Y	0.6	1	7	0.1	0.7
P9	Y	0.55	1	9	0.1	0.9
P10	Y	0.55	1	9	0.1	0.9
P11	N	0.3	2	9	0.2	0.9
P12	N	0.25	3	9	0.3	0.9
P13	Y	0.25	3	10	0.3	1
P14	N	0.2	4	10	0.4	1
P15	N	0.1	7	10	0.7	1
P16	N	0.1	7	10	0.7	1
P17	N	0.1	7	10	0.7	1
P18	N	0	8	10	0.8	1
P19	N	0	9	10	0.9	1
P20	N	0	10	10	1	1



ROC curve and **AUC**

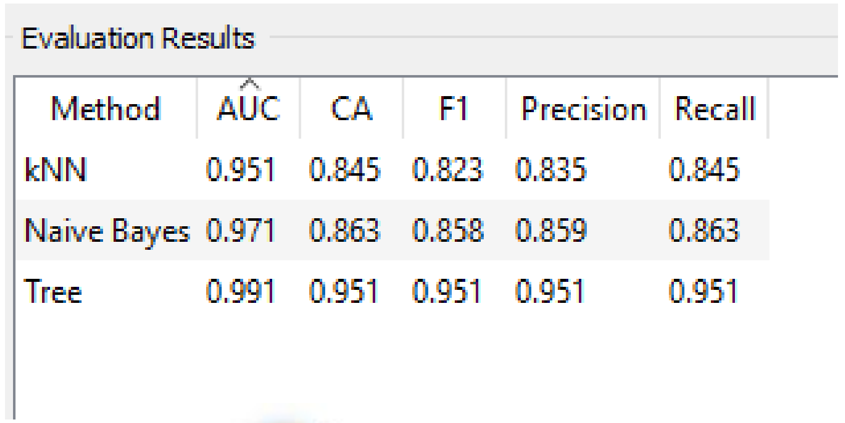
	Actual class	Classifier confidence forclass Y	FP	TP	FPr	TPr
P1	Y	1	0	2	0	0.2
P2	Y	1	0	2	0	0.2
P3	Y	0.95	0	3	0	0.3
P4	Y	0.9	0	5	0	0.5
P5	Y	0.9	0	5	0	0.5
P6	N	0.85	1	5	0.1	0.5
P7	Y	0.8	1	6	0.1	0.6
P8	Y	0.6	1	7	0.1	0.7
P9	Y	0.55	1	9	0.1	0.9
P10	Y	0.55	1	9	0.1	0.9
P11	N	0.3	2	9	0.2	0.9
P12	N	0.25	3	9	0.3	0.9
P13	Y	0.25	3	10	0.3	1
P14	N	0.2	4	10	0.4	1
P15	N	0.1	7	10	0.7	1
P16	N	0.1	7	10	0.7	1
P17	N	0.1	7	10	0.7	1
P18	N	0	8	10	0.8	1
P19	N	0	9	10	0.9	1
P20	N	0	10	10	1	1



Area Under (the convex) Curve
AUC = 0.96

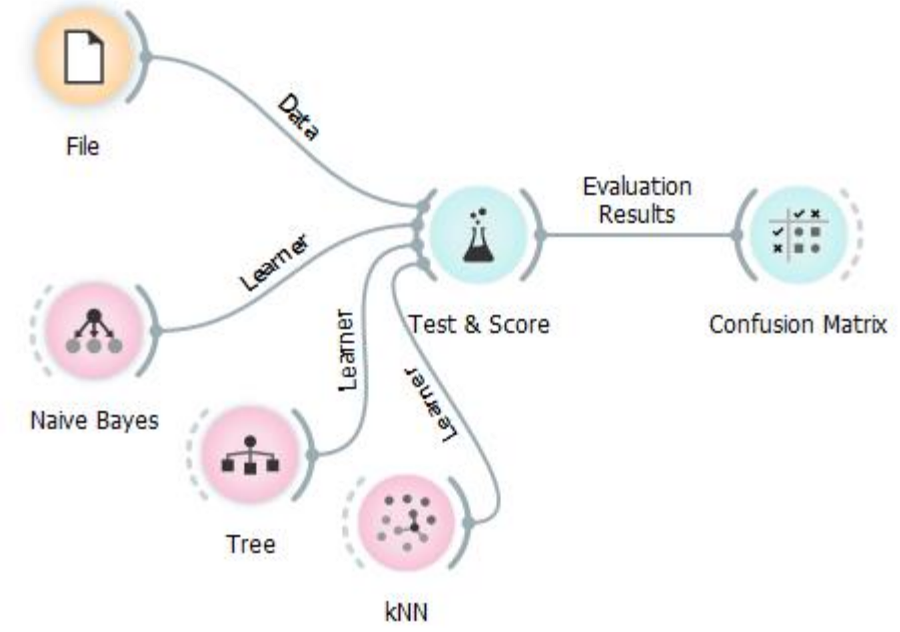
Classification evaluation in Orange

- AUC
  - Area under curve
  - AUROC
  - Površina pod ROC krivuljo
- CA – classification accuracy
  - Klasifikacijska točnost
- F1 – harmonično povprečje priklica in natančnosti
- Precision – natančnost
- Recall - priklic

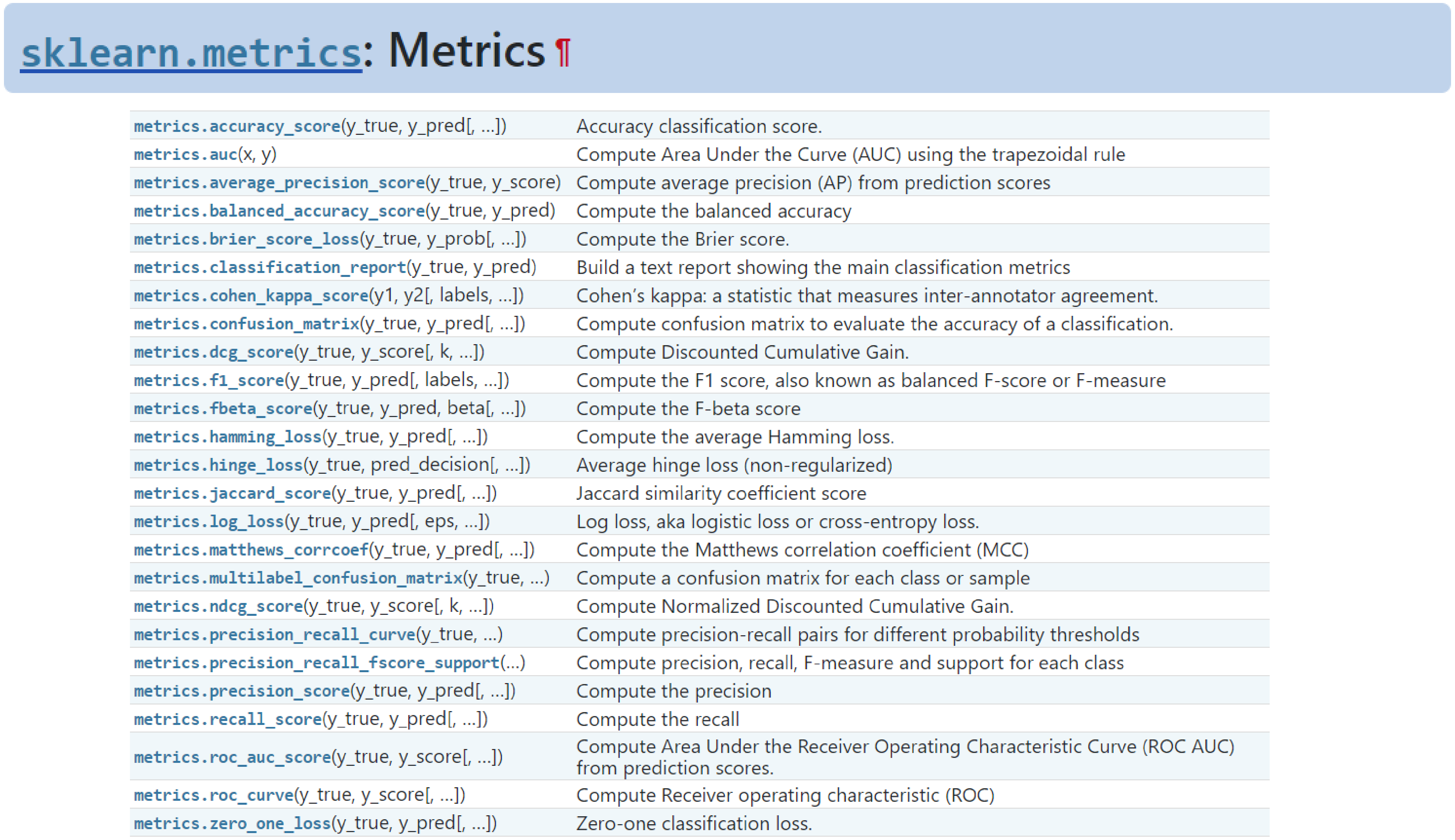


Evaluation Results

Method	AUC	CA	F1	Precision	Recall
kNN	0.951	0.845	0.823	0.835	0.845
Naive Bayes	0.971	0.863	0.858	0.859	0.863
Tree	0.991	0.951	0.951	0.951	0.951



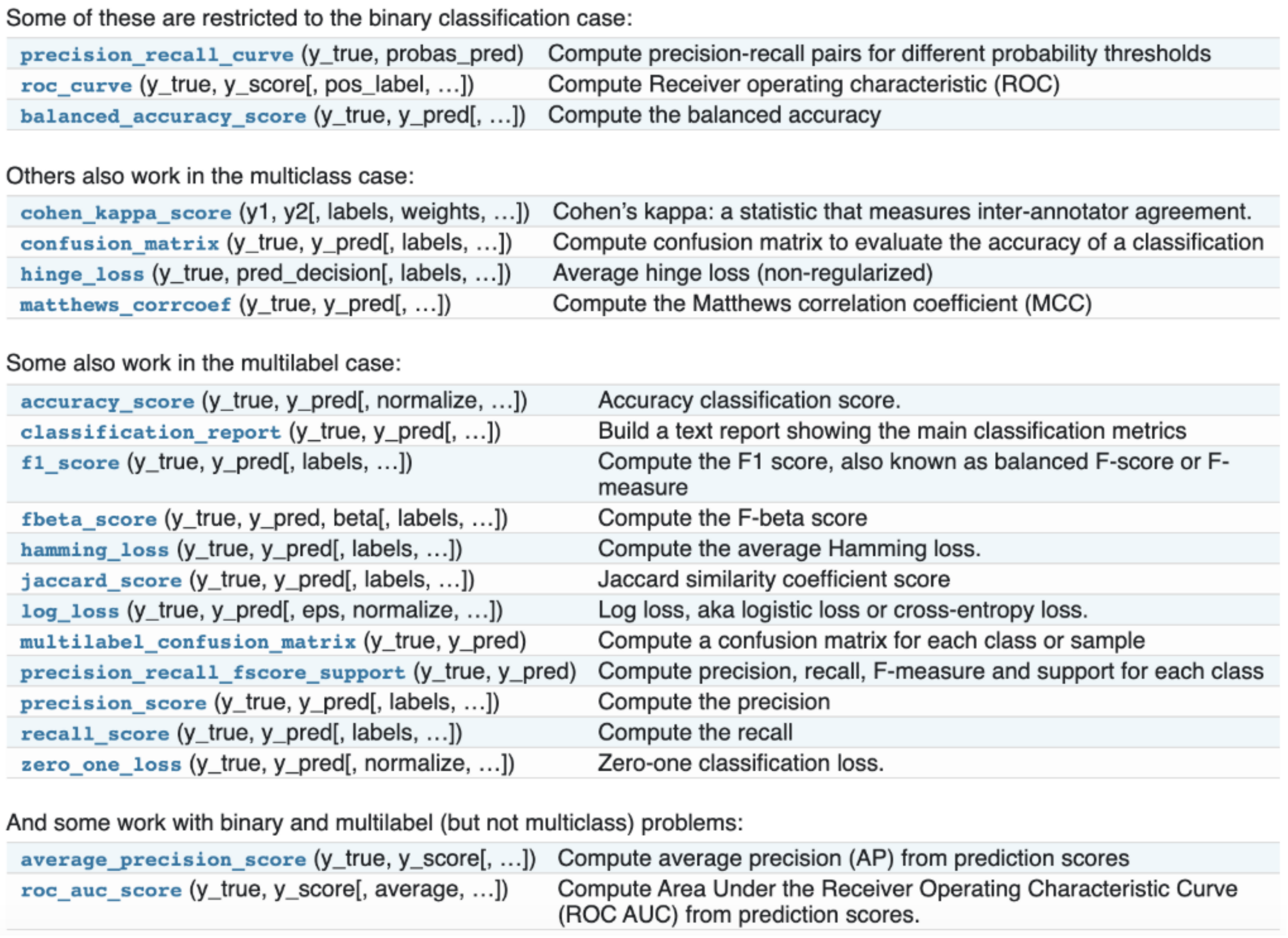
https://scikit-learn.org/stable/modules/classes.html#sklearn-metrics-metrics



sklearn.metrics: Metrics ¶

`metrics.accuracy_score`(y_true, y_pred[, …])	Accuracy classification score.
`metrics.auc`(x, y)	Compute Area Under the Curve (AUC) using the trapezoidal rule
`metrics.average_precision_score`(y_true, y_score)	Compute average precision (AP) from prediction scores
`metrics.balanced_accuracy_score`(y_true, y_pred)	Compute the balanced accuracy
`metrics.brier_score_loss`(y_true, y_prob[, …])	Compute the Brier score.
`metrics.classification_report`(y_true, y_pred)	Build a text report showing the main classification metrics
`metrics.cohen_kappa_score`(y1, y2[, labels, …])	Cohen's kappa: a statistic that measures inter-annotator agreement.
`metrics.confusion_matrix`(y_true, y_pred[, …])	Compute confusion matrix to evaluate the accuracy of a classification.
`metrics.dcg_score`(y_true, y_score[, k, …])	Compute Discounted Cumulative Gain.
`metrics.f1_score`(y_true, y_pred[, labels, …])	Compute the F1 score, also known as balanced F-score or F-measure
`metrics.fbeta_score`(y_true, y_pred, beta[, …])	Compute the F-beta score
`metrics.hamming_loss`(y_true, y_pred[, …])	Compute the average Hamming loss.
`metrics.hinge_loss`(y_true, pred_decision[, …])	Average hinge loss (non-regularized)
`metrics.jaccard_score`(y_true, y_pred[, …])	Jaccard similarity coefficient score
`metrics.log_loss`(y_true, y_pred[, eps, …])	Log loss, aka logistic loss or cross-entropy loss.
`metrics.matthews_corrcoef`(y_true, y_pred[, …])	Compute the Matthews correlation coefficient (MCC)
`metrics.multilabel_confusion_matrix`(y_true, …)	Compute a confusion matrix for each class or sample
`metrics.ndcg_score`(y_true, y_score[, k, …])	Compute Normalized Discounted Cumulative Gain.
`metrics.precision_recall_curve`(y_true, …)	Compute precision-recall pairs for different probability thresholds
`metrics.precision_recall_fscore_support`(…)	Compute precision, recall, F-measure and support for each class
`metrics.precision_score`(y_true, y_pred[, …])	Compute the precision
`metrics.recall_score`(y_true, y_pred[, …])	Compute the recall
`metrics.roc_auc_score`(y_true, y_score[, …])	Compute Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.
`metrics.roc_curve`(y_true, y_score[, …])	Compute Receiver operating characteristic (ROC)
`metrics.zero_one_loss`(y_true, y_pred[, …])	Zero-one classification loss.

21 measures of accuracy from scikit-learn documentation for Classification problems



Some of these are restricted to the binary classification case:

precision_recall_curve (y_true, probas_pred)	Compute precision-recall pairs for different probability thresholds
roc_curve (y_true, y_score[, pos_label, …])	Compute Receiver operating characteristic (ROC)
balanced_accuracy_score (y_true, y_pred[, …])	Compute the balanced accuracy

Others also work in the multiclass case:

cohen_kappa_score (y1, y2[, labels, weights, …])	Cohen's kappa: a statistic that measures inter-annotator agreement.
confusion_matrix (y_true, y_pred[, labels, …])	Compute confusion matrix to evaluate the accuracy of a classification
hinge_loss (y_true, pred_decision[, labels, …])	Average hinge loss (non-regularized)
matthews_corrcoef (y_true, y_pred[, …])	Compute the Matthews correlation coefficient (MCC)

Some also work in the multilabel case:

accuracy_score (y_true, y_pred[, normalize, …])	Accuracy classification score.
classification_report (y_true, y_pred[, …])	Build a text report showing the main classification metrics
f1_score (y_true, y_pred[, labels, …])	Compute the F1 score, also known as balanced F-score or F-measure
fbeta_score (y_true, y_pred, beta[, labels, …])	Compute the F-beta score
hamming_loss (y_true, y_pred[, labels, …])	Compute the average Hamming loss.
jaccard_score (y_true, y_pred[, labels, …])	Jaccard similarity coefficient score
log_loss (y_true, y_pred[, eps, normalize, …])	Log loss, aka logistic loss or cross-entropy loss.
multilabel_confusion_matrix (y_true, y_pred)	Compute a confusion matrix for each class or sample
precision_recall_fscore_support (y_true, y_pred)	Compute precision, recall, F-measure and support for each class
precision_score (y_true, y_pred[, labels, …])	Compute the precision
recall_score (y_true, y_pred[, labels, …])	Compute the recall
zero_one_loss (y_true, y_pred[, normalize, …])	Zero-one classification loss.

And some work with binary and multilabel (but not multiclass) problems:

average_precision_score (y_true, y_score[, …])	Compute average precision (AP) from prediction scores
roc_auc_score (y_true, y_score[, average, …])	Compute Area Under the Receiver Operating Characteristic Curve (ROC AUC) from prediction scores.

21 measures of accuracy from scikit-learn documentation for Classification problems

Félix Revert: The proper way to use Machine Learning metrics
https://towardsdatascience.com/the-proper-way-to-use-machine-learning-metrics-4803247a2578

Probabilistic classification

A **probabilistic** classifier is a classifier that is able to predict, given an observation of an input, a **probability** distribution over a set of classes, rather than only outputting the most likely class that the observation should belong to.

$$\boxed{p(C_k \mid x_1, \ldots, x_n)}$$

The idea behind the Naïve Bayes Classifier

- We are interested in the probability of the class C given the attribute values X_1, X_2, X_3, …. , X_n

$$P(C|X_1 X_2 \ldots X_n)$$

- We „**naively**“ assume that all attribute values X_1, X_2, X_3, …. , X_n are mutually independent, conditional on the category C

$$P(X_1 X_2 \ldots X_n|C) \approx P(X_1|C) \cdot P(X_2|C) \cdot \ldots \cdot P(X_n|C)$$

$$\begin{aligned} p(C_k, x_1, \dots, x_n) &= p(x_1, \dots, x_n, C_k) \\ &= p(x_1 \mid x_2, \dots, x_n, C_k)\, p(x_2, \dots, x_n, C_k) \\ &= p(x_1 \mid x_2, \dots, x_n, C_k)\, p(x_2 \mid x_3, \dots, x_n, C_k)\, p(x_3, \dots, x_n, C_k) \\ &= \dots \\ &= p(x_1 \mid x_2, \dots, x_n, C_k)\, p(x_2 \mid x_3, \dots, x_n, C_k) \dots p(x_{n-1} \mid x_n, C_k)\, p(x_n \mid C_k)\, p(C_k) \end{aligned}$$

Now the "naive" conditional independence assumptions come into play: assume that all features in $\mathbf{x}$ are mutually independent, conditional on the category C_k. Under this assumption,

$$p(x_i \mid x_{i+1}, \dots, x_n, C_k) = p(x_i \mid C_k)\,.$$

Thus, the joint model can be expressed as

$$\begin{aligned} p(C_k \mid x_1, \dots, x_n) &\propto p(C_k, x_1, \dots, x_n) \\ &= p(C_k)\, p(x_1 \mid C_k)\, p(x_2 \mid C_k)\, p(x_3 \mid C_k)\, \cdots \\ &= p(C_k) \prod_{i=1}^{n} p(x_i \mid C_k)\,, \end{aligned}$$

Naïve Bayes Classifier

Conditional probability of attribute value v_i given class c

Attribute values

$$P(c_i \mid a_1 = v_1, a_2 = v_2, \ldots, a_j = v_j) \propto P(c_i) \times \prod_{j=1}^{n} \overline{P(a_j = v_j \mid class = c_i)}$$

Class c_i

* where ∝ denotes proportionality
* The results are not probabilities (they do not sum up to 1). The formula is simplified for easy implementation (and time complexity), while the results are proportional to the estimates of the probabilities of a class given the attribute values.

Naïve Bayes Classifier

Conditional probability of attribute value x_i given class Ck

$$\hat{y} = \underset{k \in \{1,\dots,K\}}{\operatorname{argmax}} \; p(C_k) \prod_{i=1}^{n} \overline{p(x_i \mid C_k)}$$

Predicted class

- k is the number of all classes
- Ck are the classes
- xi are the attribute values
- The selected class is the one with the maximum of the

Home reading

Read: Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The elements of statistical learning, Second edition*. New York: Springer series in statistics. https://web.stanford.edu/~hastie/Papers/ESLII.pdf

Pages 9 – 18: